

# NAVAL POSTGRADUATE SCHOOL MONTEREY, CALIFORNIA



## PERFORMANCE ANALYSIS OF BORDER GATEWAY PROTOCOL (BGP-4)

by

Douglas A. Powers  
LT                      USN

March 22, 1999

**Prepared for:**  
**Professor John McEachen**  
**AY99 Winter Quarter**  
**EC4800 Final Project**

## **TABLE OF CONTENTS**

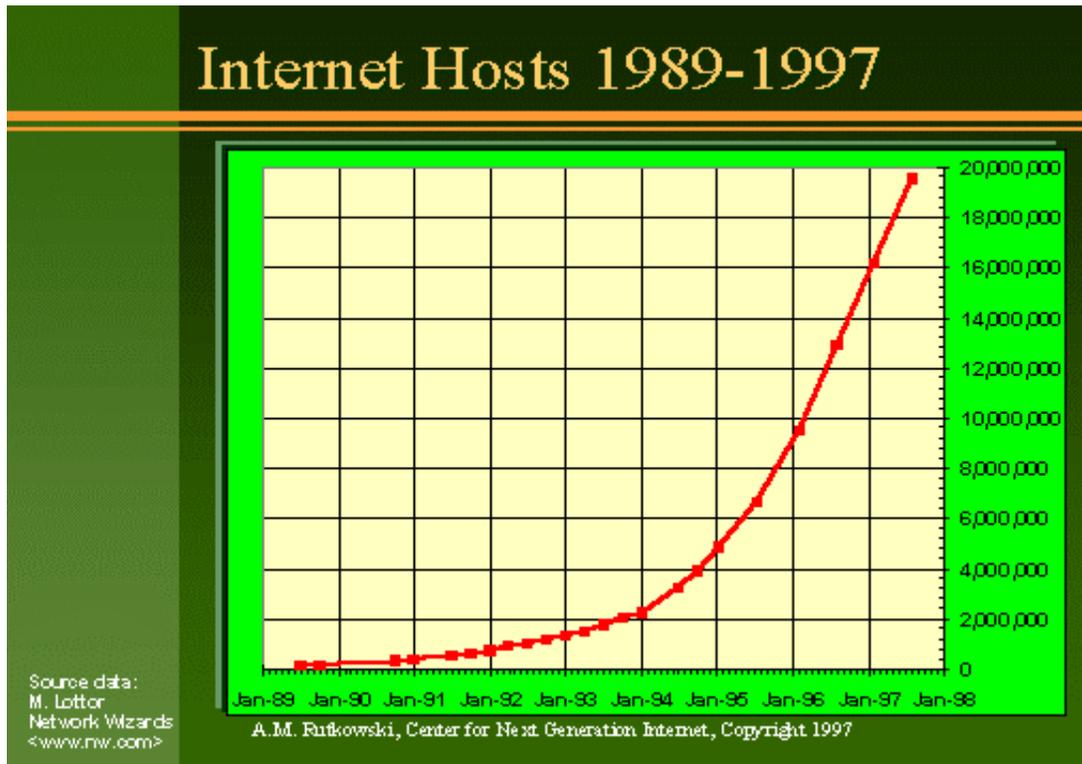
I.	INTRODUCTION .....	3
II.	BACKGROUND OF BORDER GATEWAY PROTOCOL (BGP).....	7
	A. HISTORY AND APPLICABLE DOCUMENTATION.....	7
	B. SUBNETTING AND CLASSLESS INTER-DOMAIN ROUTING (CIDR) .....	8
	C. SUPERNETTING.....	10
	D. HOP-BY-HOP PARADIGM.....	12
	E. TRANSPORT LAYER DEPENDANCY.....	13
III.	BGP OPERATION AND ARCHITECTURE.....	15
	A. GENERAL OVERVIEW.....	15
	B. BASIC ARCHITECTURE.....	15
	C. ROUTE ADVERTISEMENT AND STORAGE.....	17
	D. BGP FINITE STATE MACHINE (FSM) .....	18
	E. BGP MESSAGE FORMATS.....	23
	F. BGP METRICS AND PATH ATTRIBUTES .....	33
	G. BGP ALGORITHM.....	38
IV.	PROTOCOL ANALYSIS .....	40
	A. KEY FEATURES .....	40
	B. PERFORMANCE .....	43
V.	CONCLUSION.....	52
	LIST OF REFERENCES.....	53

## LIST OF FIGURES

FIGURE 1: Internet Growth 1989-1997 .....	3
FIGURE 2: Internet Growth Trends .....	4
FIGURE 3: BGP is a Transport Layer Protocol .....	13
FIGURE 4: Example BGP Packet Encapsulation .....	14
FIGURE 5: BGP Six-State FSM .....	18
FIGURE 6: BGP Header and Four Message Types .....	23
FIGURE 7: BGP Header Format .....	24
FIGURE 8: BGP OPEN Message Format .....	26
FIGURE 9: BGP UPDATE Message Format.....	29
FIGURE 10: BGP NOTIFICATION Message Format .....	31
FIGURE 11: BGP KEEP-ALIVE Message Format .....	33
FIGURE 12: BGP Algorithm.....	38
FIGURE 13: BGP Bandwidth Consumption.....	44
FIGURE 14: Internet BGP Prefixes 1994-Present .....	45
FIGURE 15: BGP Router Memory Requirements (MR) .....	50

## I. INTRODUCTION

The Internet has continued to grow at an exponential in the last few years and it has already begun to face some serious scaling issues. As early as 1995, the growth of routing tables in Internet routers were starting to expand beyond the ability of the most software to effectively manage (see Figure 1 below). The most widely used software at the time was a gateway routing protocol known as the Exterior Gateway Protocol (EGP) which was designed to work with the original ARPANET. Operational experience with EGP had shown that it had soon become highly inadequate for rapidly the Internet. The main inadequacy was the ability for the protocol to address needed route aggregation methods and the need for Internet gateway routers to support subnetting.

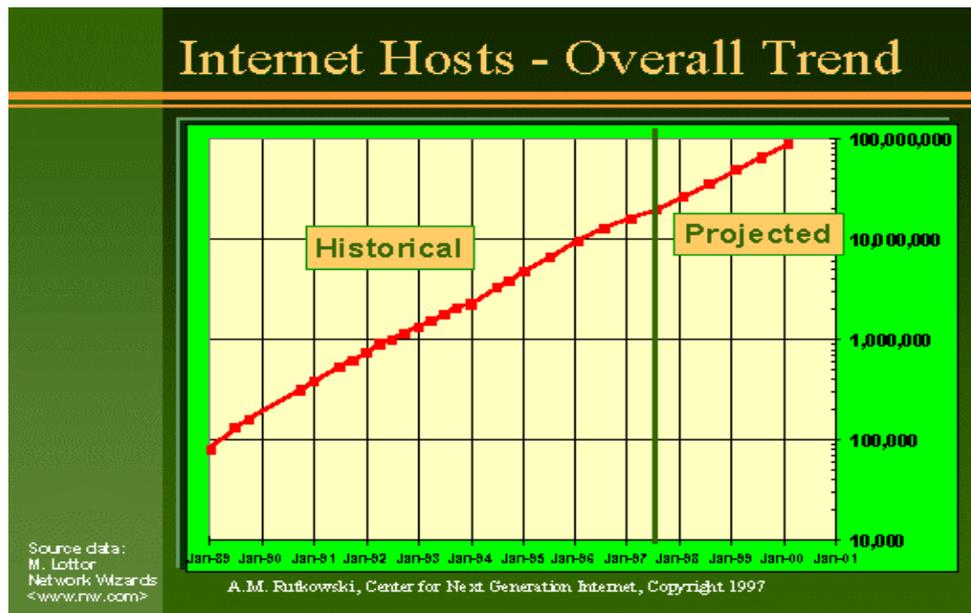


**Figure 1: Internet Growth 1989-1997**

(Source: Network Wizards, <http://www.nw.com>)

Another scaling problem that had begun to occur was the rapid depletion of the Class-B network address space. One fundamental cause of this problem was the lack of a network class of a size that is appropriate for mid-sized organizations, which is the largest class size needed. For example, a class-C address, with a maximum of 254 host addresses, is too small for a mid-size organization, while a class-B address, which allows up to 65534 addresses, is too large to be densely populated, and therefore, wastes valuable address space (Rekhter, 1995).

The use of the Border Gateway Protocol (BGP), in combination with Classless Inter-Domain Routing (CIDR), has been a good interim solution for both of these problems. Together, BGP and CIDR have implemented mechanisms to slow the growth of the Internet routing table and to slow the need for allocating new Internet Protocol (IP) network numbers. The size of the Internet cannot continue to double every year forever, but this growth rate is expected to continue at least through the year 2001 (see Figure 2).



**Figure 2: Internet Growth Trends**

(Source: Network Wizards, <http://www.nw.com>)

There is a third scaling problem that BGP has assisted in easing the burden but it is one that it does not solve. This concern is the eventual exhaustion of the 32-bit IP addresses, which is rapidly approaching. BGP has simply addressed the first two scaling issues, which have been more immediate concerns over the past few years. The implementation of IP version 6 (IPv6) directly addresses this third scaling issue, by introducing an abundance of new IP address for the global Internet.

Meanwhile, BGP has been very effective in allowing the Internet to continue to grow at a rapid rate while maintaining efficient functionality (Rekhter, 1995). In fact, BGP is not only well suited for the current Internet, it could be viewed as a necessity for the current Internet as well.

This paper contains three main sections to follow. Section II introduces the current version of the internetworking protocol, its origin and brief history, the current applicable information related to BGP-4 (which is readily available on the Internet at a variety of locations) and gives a basic overview of the concepts of subnetting, CIDR, and supernetting. A clear understanding of these concepts are crucial to one realizing the inter-workings of BGP. BGP's routing paradigm and its reliance on and interaction with the Transport Control Protocol (TCP) are also introduced in this section. Readers knowledgeable in the inter-networking field may find this section somewhat elementary and therefore, may want to move directly to Section III.

Section III is a cursory overview of BGP's operation and architecture. Specifically, this section includes coverage on the protocol's method of route advertisement and storage, the BGP Finite State Machine (FSM) model, its message formats, some its important metrics and its routing algorithm. This section is only

intended to be a general reference section with which most of its content is a summary of the very detailed coverage found in several of the current standards documents for the protocol, the RFCs. This section is intended to provide the reader with enough working knowledge and familiarity with BGP in order to understand the engineering observations and protocol analysis comments included in Section IV, which is the main focus of this paper.

## **II. BACKGROUND OF BORDER GATEWAY PROTOCOL (BGP)**

### **A. HISTORY AND APPLICABLE DOCUMENTATION**

Border Gateway Protocol (BGP) is an inter-Autonomous System (AS) routing protocol designed for Transfer Control Protocol/Internet Protocol (TCP/IP) internets. While BGP has been used in the production environment since 1989, it didn't become the most widely used inter-AS protocol until only a few years ago. BGP-4, which is the current version as well as the current de-facto Internet standard, is described in Request For Comments (RFC) 1771, which was published in March 1995.

BGP-4 is an extension of BGP-3 that provides support for routing information aggregation and reduction based on the Classless Inter-Domain Routing (CIDR) architecture. BGP was designed based on experience gained with the Exterior Gateway Protocol (EGP) as defined in RFC 904 and EGP usage in the NSFNET Backbone as described in RFC 1092 and RFC 1093. Version 1 of the BGP protocol was published in RFC 1105 in 1989. Since then, BGP versions 2, 3, and 4 were developed with corresponding RFC's published in 1990, 1991 and 1995, respectively. Version 2 is documented in RFC 1163 and version 3 in RFC 1267. All of the functionality that was present in the previous versions is present in version 4 (Rekhter, 1995).

RFC 1771 defines an Autonomous System as a set of routers under a single technical administration, using an interior gateway protocol's (IGP) common metrics to route packets within the AS, and using an exterior gateway protocol to route packets to other ASs. This description is now viewed as the classic definition of an AS since some changes in actual AS architecture have since developed that differ from this original definition. Specifically, in recent years it has become common for a single AS to use

multiple IGPs and sometimes several sets of metrics within a single AS. The key requirement for a network to be an AS is that regardless of the number of IGPs and metrics it uses, its administration of itself must appear to other ASs to have a single coherent interior routing plan. It also must present a consistent picture of what destinations are reachable through it (Rekhter, 1995).

BGP's main function is to act as a speaking system between ASs in an effort to exchange network routing and coordination information. More specifically, the BGP speaking system allows ASs to share reachability information of other ASs throughout the entire Internet. The network reachability information is used to construct a graph, or BGP routing table, of the connectivity of all ASs. Specifically, this information exchanged between BGP speakers contains full AS paths that can be used to implement local AS policy decisions, prune routing loops and enforce routing policies based on performance preferences, policy constraints and configuration choices (Rekhter, 1995).

## **B. SUBNETTING AND CLASSLESS INTER-DOMAIN ROUTING (CIDR)**

One of the greatest advantages of BGP-4 is its ability to support Classless Inter-Domain Routing (CIDR). Prior to BGP-4, BGP-3 (as well as EGP) required an AS to advertise externally every single network Class within its own AS. BGP-4 incorporates mechanisms that allow BGP speakers to simplify their list of advertisement information by only advertising Internet Protocol (IP) *prefixes* to other ASs. An IP address is a 32-bit number consisting of 4 bytes, called octets, and its IP *prefix* is the minimum decimal number of high-order bits needed to uniquely identify it. Prefixes can also be thought of as 'routes' to a certain physical locale. A commonly used notation in the internetworking

community is to write the first two octets of an address in decimal followed by a slash (/) and the prefix size in decimal (i.e., the Class B address of 131.20/24, has a 24-bit prefix in this case). A Class B address would normally have a 24-bit prefix unless subnetting is implemented, whereby the prefix would be longer.

By advertising IP prefixes, or routes, BGP takes advantage of subnetting. Subnetting occurs when one address prefix, which corresponds to a physical locale, is extended into longer sub-prefixes which correspond to smaller physical locales. With CIDR, it is not necessary to advertise every single sub-prefix to other AS's, when their general locale can be advertised by pointing to the main prefix (or route) where they are located. Subnetting allows a network administrator to implement the maximum number and combination of subnets and hosts appropriate for his network.

**Example:**

**If a network administrator is responsible for administering the IP network at the Naval Postgraduate School (NPS), he may choose to use an address prefix for the entire campus, such as the B Class address of 131.120/16. Next, he chooses to subnet this prefix into longer prefixes for other buildings within the campus. Perhaps he assigns 131.120.1/24 to Spanegal Hall, 131.120.2/24 to Glasgow Hall, 131.120.3/24 for Hermann Hall, etc. However, it can get more complicated. Spanegal Hall may have 300 computers (or hosts) in it. A 24-bit prefix, which only leaves room for 256 host addresses (254 assignable), won't work. So he chooses to use 131.120.2/23 for Spanegal Hall, which means 131.120.3/24 won't be available for Hermann Hall, since the prefixes overlap. Subnetting must be carefully planned to**

**properly allow for the correct number of hosts to each subnet. Ultimately, the NPS campus buildings are interconnected with routers, which use the prefixes to direct traffic among the buildings. The real advantage of CIDR and consequently BGP (if implemented) is demonstrated when the routers connecting the campus to outside networks only need to advertise a single prefix (or route), 131.120/16, for the rest of the Internet to access the campus gateway.**

Not only does CIDR simplify the path resolution to other ASs throughout the Internet, it reduces the required size of routing tables and BGP tables on every BGP speaker.

### **C. SUPERNETTING**

Most of the bandwidth on internet routing tables is consumed by the exchange of the Network Layer Reachability Information (NLRI). BGP-4 was created specifically to reduce the amount of NLRI entries carried and exchanged by border routers. BGP-4, along with CIDR, has introduced the concept of “supernetting”, which describes a power-of-two aggregation of more than one Class-based network.

BGP-4 supports the concept of supernetting by allowing for aggregation of routes (prefixes), including the aggregation of AS paths. Supernetting (RFC 1518 and RFC 1519) occurs when multiple network addresses of the same Class are combined into blocks. The requirements for supernetting are that the network addresses must be consecutive and that they must fall on the correct boundaries. For example, to combine two Class C networks, they must be consecutive, and the third octet of the first address

must be divisible by two. In order to supernet eight Class C addresses, they must be consecutive, and the third octet of the first address must be divisible by 8. For example, **131.103.15.0** and **131.103.16.0** cannot be combined into a supernet, but **131.103.18.0** and **131.103.19.0** may.

Similar to the previous example, supernetting is most often used to combine Class C addresses. A single Class C IP network has 24 bits for the network portion, and eight bits for the host portion of the IP address. This gives a possibility of 256 hosts within a Class C IP network.

The subnet mask for a Class C is **255.255.255.0**, or 24 high-order bits of ones. In order to supernet, the number of bits used for the subnet mask needs to be reduced. By using a 23-bit mask, **255.255.254.0**, 23 bits are allocated for the network portion, and nine bits for the host portion. A single IP network with 512 addresses is created.

**Example:**

**Take two Class C networks of 131.103.78.0 and 131.103.79.0. The addresses pass the requirements. They are consecutive and the third octet of the first address is divisible by two. Consider the addresses in binary. Decimal 78 is binary 01001110. Decimal 79 is 01001111. The binaries are the same except for the last bit, which corresponds to the 24th bit of the IP address. The 78 network is referred to as supernet 0 and the 79 network is supernet 1.**

**The subnet mask for this example supernet is 23 bits, or 255.255.254.0. All devices on the network must use this subnet mask. Any device not using this subnet mask is unreachable.**

**The network address for this supernet is 131.103.78.0, and the broadcast address is 131.103.79.255. The network address is used as the destination address for routes to this network. The broadcast address is used as a special destination for all hosts on the network. The network and broadcast addresses are reserved and may not be applied to any device.**

Because of these unique addresses, it would probably be wise not to use the **131.103.78.255** and **131.103.79.0** addresses in the above example, even though these are legal addresses for hosts when using this supernet. With supernetting, either static routes or RIP version 2 (which, like BGP-4, supports CIDR) must be used, since there is a high probability of encountering variable subnet masks.

#### **D. HOP-BY-HOP PARADIGM**

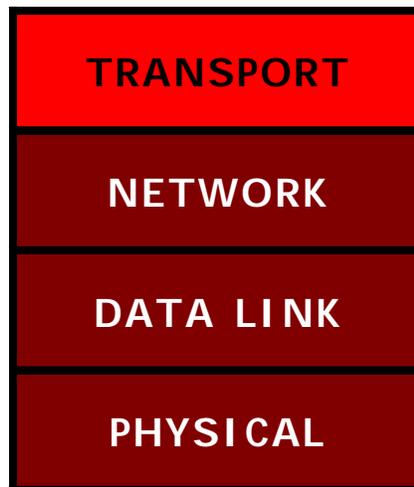
In addition to being a good solution to some of the Internet's scaling issues, BGP's routing paradigm is also appropriate for the current Internet. BGP uses a "hop-by-hop" paradigm as do most other routing protocols, however, it is slightly different. BGP views the "next hop" as the next AS, not the next router.

This concept is an important one to understand since it is the basis by which one characterizes the set of policy decisions that can be enforced using BGP. One must focus on the rule that a BGP speaker (a border router that implements BGP) advertise to its peers (other BGP speakers which it communicates with) in neighboring ASs only those routes that it itself uses. This rule is a reflection of the "hop-by-hop" routing paradigm. Some policies cannot be supported by the "hop-by-hop" routing

paradigm and thus require techniques such as source routing for policy enforcement. For example, BGP does not enable one AS to send traffic to a neighboring AS intending that the traffic take a different route from that taken by traffic originating in the neighboring AS. Since the current Internet uses only the "hop-by-hop" routing paradigm and since BGP can support any policy that conforms to that paradigm, BGP is highly applicable as an inter-AS routing protocol for the current Internet.

### **E. TRANSPORT LAYER DEPENDENCY**

BGP was designed to operate at the transport layer over a reliable transport protocol, which for the current Internet is the Transfer Control Protocol (TCP). See Figures 3 and 4. TCP meets BGP's transport requirements and is present in virtually all commercial routers and hosts. TCP operates on port 179 for establishing its connections.



**Figure 3: BGP is a Transport Layer Protocol**

Running over TCP is a great advantage in regards to simplifying BGP's implementation. By relying on TCP, BGP eliminates its need to implement explicit

update fragmentation, retransmission, acknowledgement, and sequencing. Any authentication scheme used by TCP may be used in addition to BGP's own authentication mechanisms.



**Figure 4: Example BGP Packet Encapsulation**

This dependency on TCP also inherits weaknesses. The error notification mechanism used in BGP assumes that TCP supports a graceful close, meaning that all outstanding data will be delivered before the connection is closed. Therefore, an error in TCP, is an error for BGP. Likewise, any security vulnerability that exists with TCP also poses a threat to BGP.

### III. BGP OPERATION AND ARCHITECTURE

#### A. GENERAL OVERVIEW

Although the specifics of how BGP operates can be somewhat complicated, the basic overview of BGP operation is rather simple. BGP communication begins when two systems form a TCP connection between one another by using the BGP **OPEN** message. They exchange messages to open and confirm the connection parameters. The initial data flow is the entire BGP routing table, then incremental updates, in the form of **UPDATE** messages, are sent as the routing tables change in time. A noteworthy characteristic of this protocol is that it does not require a periodic refresh of the entire BGP routing table. Consequently, a BGP speaker must retain the current version of the entire BGP routing tables of all of its peers for the duration of the connection. **KEEP\_ALIVE** messages are sent periodically to ensure the liveliness of the connection. **NOTIFICATION** messages are sent in response to errors or special conditions. If a connection encounters an error condition, a **NOTIFICATION** message is sent and the TCP connection is closed.

#### B. BASIC ARCHITECTURE

As mentioned previously, a BGP system exchanges NLRI with other BGP systems. Some of this network reachability information is *local* traffic and some of it is *transit* traffic. A major goal of BGP usage in the Internet has been to reduce transit traffic. An AS can be categorized as either a *stub*, a *multi-homed* or *transit* AS. A stub AS has only a single connection to one other AS and only carries local traffic. A multi-homed AS has connections to more than one other AS, but refuses to carry transit traffic.

A transit AS has connections to more than one other AS and is designed, under certain policy restrictions, to carry both local and transit traffic (Stevens, 1996).

In the case of a transit AS, there may exist multiple BGP speakers that provide transit service for other ASs. In this situation, care must be taken to ensure a consistent view of routing within the AS. A consistent view of the interior routes of the AS is provided by the IGP. A consistent view of the routes exterior to the AS can be provided by having all BGP speakers within the AS maintain direct BGP connections with each other. This intra-AS BGP communication between BGP speakers within the same AS is what is known as *internal* BGP or iBGP. Similarly, connections between BGP speakers are called internal links and a BGP speaker within in the same AS may be described as an internal peer. Using a common set of policies, the iBGP peers arrive at an agreement as to which border routers will serve as exit and entry points for particular destinations outside the AS. This information is communicated to the AS's other internal routers via the IGP. Care must be taken to ensure that the interior routers have all been updated with transit information before the BGP speakers announce to other ASs that transit service is being provided.

Connections between BGP speakers of different ASs are referred to as *external* links, and the communication architecture as external BGP or eBGP. Likewise, a BGP peer in a different AS is referred to as an external peer.

### C. ROUTE ADVERTISEMENT AND STORAGE

RFC 1771 defines a route as a unit of information that pairs a destination with the attributes of a path to that destination (Rekhter, 1995). Routes are advertised between a pair of BGP speakers in **UPDATE** messages and are stored in Routing Information Bases (RIBs). RIBs have three parts; the Adj-RIBs-In, the Loc-RIB, and the Adj-RIBs-Out. Routes that are received from other BGP speakers are present in the Adj-RIBs-In. Routes that will be used by the local BGP speaker must be present in the Loc-RIB, and routes that will be advertised to other BGP speakers must be present in the Adj-RIB-Out. The next hop for each of these routes must be present in the local BGP speaker's forwarding information base as well.

When a BGP speaker chooses to advertise a route, it may add to or modify the path attributes of the route before advertising it to a peer. BGP provides mechanisms by which a BGP speaker can inform its peer that a previously advertised route is no longer available for use. There are three such methods by which a given BGP speaker can indicate that a route has been withdrawn from service:

- a) The IP prefix that expresses destinations for a previously advertised route can be advertised in the **WITHDRAWN ROUTES** field in the **UPDATE** message, thus marking the associated route as being no longer available for use.
- b) A replacement route with the same NLRI can be advertised, or
- c) The BGP speaker to BGP speaker connection can be closed, which implicitly removes from service all routes which the pair of speakers had previously advertised to each other.

## D. BGP FINITE STATE MACHINE (FSM)

A more thorough understanding of how BGP operates can be gained from an engineering standpoint by viewing the BGP operation as a Finite State Machine (FSM). The BGP FSM has six distinct states. These states are *Idle*, *Connect*, *Active*, *OpenSent*, *OpenConfirm* and *Established* (Figure 5). There are also thirteen *events* (manual or automatic commands) and five *timers* that control the state transitions. All states will be discussed here, but only a few of the events and timers need be mentioned for the level of detail of required for the purpose of this paper.

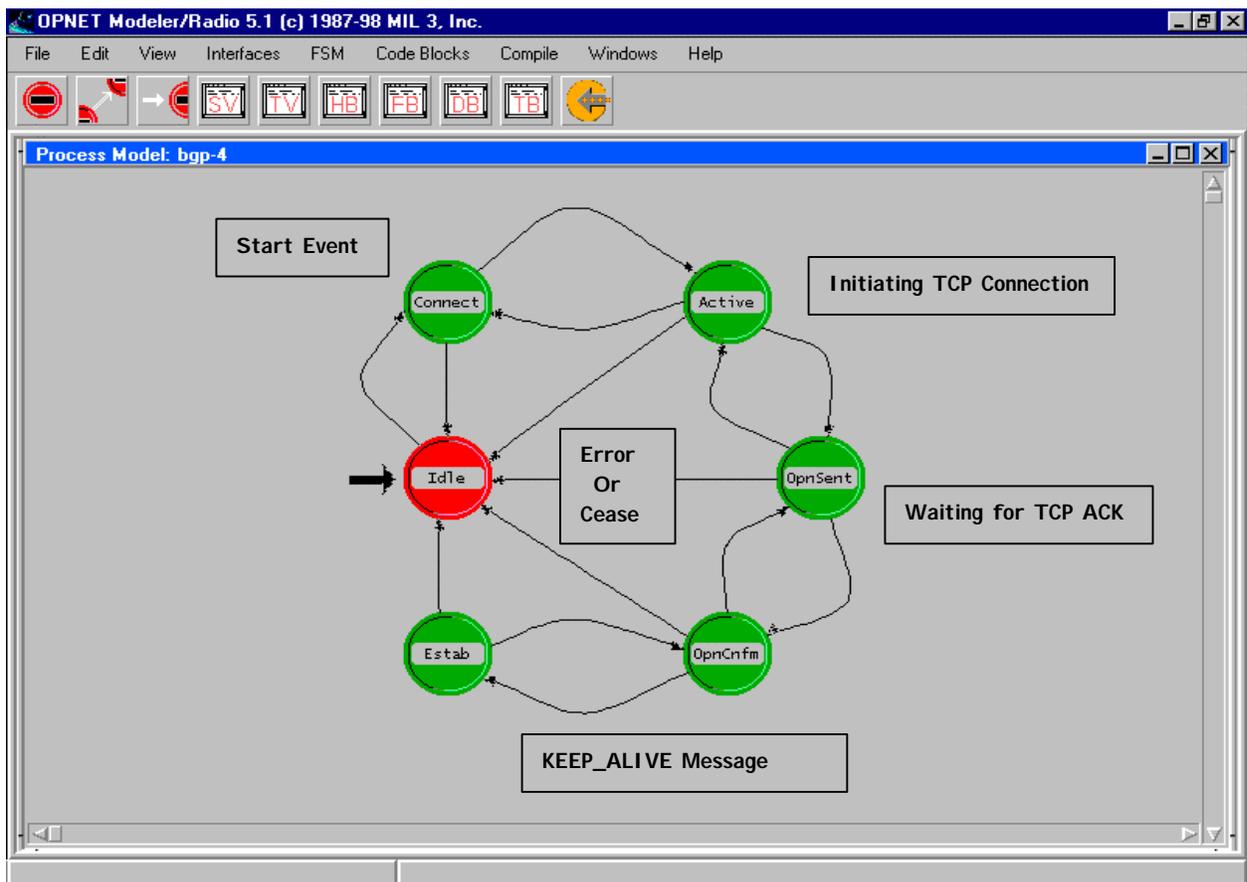


Figure 5: BGP Six-State FSM

Initially BGP is in the *Idle* state. In this state BGP refuses all incoming BGP connections, regardless of any incoming BGP over TCP correspondence. No resources are allocated to any peer(s). In response to the *Start* event (initiated by either system or operator) the local system initializes all BGP resources, starts the *ConnectRetry* timer, initiates a TCP connection to any internal BGP peers, while listening for connection that may be initiated by any external BGP peers, and changes its state to *Connect*. The exact value of the *ConnectRetry* timer is a local matter, but should be sufficiently large to allow TCP initialization. If a BGP speaker detects an error, it shuts down the connection and changes its state back to *Idle*. Any other event received in the *Idle* state is ignored (Rekhter, 1995).

In the *Connect* state, BGP is waiting for the TCP connection to be completed. Once the three-way TCP handshake is complete, the local system clears the *ConnectRetry* timer, completes initialization, sends an OPEN message to its peer (to be discussed in detail in Section III.E.), and changes its state to *OpenSent*. If the TCP connection fails (i.e., retransmission timeout), the local system restarts the *ConnectRetry* timer, continues to listen for a connection that may be initiated by the remote (external) BGP peer, and changes its state to the *Active* state. In response to the *ConnectRetry* timer expired event, the local system restarts the *ConnectRetry* timer, initiates a TCP connection to the internal BGP peer, continues to listen for a connection that may be initiated by the remote BGP peer, and stays in the *Connect* state. In response to any other event (initiated by either system or operator), the local system releases all BGP resources associated with this connection and changes its state to *Idle* (Rekhter, 1995).

In the *Active* state, BGP is trying to acquire a peer by initiating a TCP connection. If the TCP connection succeeds, the local system clears the *ConnectRetry* timer, completes initialization, sends an OPEN message to its peer, sets its *Hold Timer* to a large value (a Hold Timer value of four minutes is suggested in RFC 1771), and changes its state to *OpenSent*. In response to the *ConnectRetry* timer expired event, the local system restarts the *ConnectRetry* timer, initiates a TCP connection to the internal BGP peer, continues to listen for a connection that may be initiated by the remote BGP peer, and changes its state to *Connect*. If the local system detects that a remote peer is trying to establish a BGP connection to it, and the IP address of the remote peer is not an expected one, the local system restarts the *ConnectRetry* timer, rejects the attempted connection, continues to listen for a connection that may be initiated by the remote BGP peer, and stays in the *Active* state. The *Start* event is ignored in the *Active* state. As in the *Connect* state, in response to any other event (initiated by either system or operator), the local system releases all BGP resources associated with this connection and changes its state back to *Idle* (Rekhter, 1995).

In this *OpenSent* state, BGP waits for an OPEN message from its peer. When an OPEN message is received, all fields are checked for correctness. If the BGP algorithm detects a message header error or OPEN message error, or even a connection collision, the local system sends a NOTIFICATION message and changes its state to *Idle*. If there are no errors in the OPEN message, BGP sends a KEEPALIVE (discussed in detail in Section III.E.) message and sets a *KeepAlive* timer. The *Hold Timer*, which was originally set to a large value (see above), is replaced with a negotiated *Hold Time* value that is the lowest value between the two peers. If the negotiated *Hold Time* value is zero,

then the *Hold Time* timer and *KeepAlive* timers are not started. If the value of the AS field is the same as the local AS number, then the connection is an "internal" connection; otherwise, it is "external". Finally, the state is changed to *OpenConfirm*. If a disconnect notification is received from TCP, the local system closes the BGP connection and restarts the *ConnectRetry* timer. If local system continues to listen for a connection attempt that may be initiated by the remote BGP peer, and goes into the *Active* (Rekhter, 1995).

If the *Hold Timer* expires, the local system sends a NOTIFICATION message (discussed in detail in section III.E.) with an error code *Hold Timer Expired* and changes its state to *Idle*. In response to the *Stop* event (initiated by either system or operator) the local system sends a NOTIFICATION message with an error code *Cease* and changes its state to *Idle*. The *Start* event is ignored in the *OpenSent* state. In response to any other event the local system sends a NOTIFICATION message with an error code *Finite State Machine Error* and changes its state to *Idle*. Whenever BGP changes its state from *OpenSent* to *Idle*, it closes the BGP and hence, the TCP connection, and releases all resources associated with that connection (Rekhter, 1995).

In the *OpenConfirm* state, BGP is waiting for a KEEPALIVE or NOTIFICATION message. If the local system receives a KEEPALIVE message, it changes its state to *Established*. If the *Hold Timer* expires before a KEEPALIVE message is received, the local system sends NOTIFICATION message with an error code *Hold Timer Expired* and changes its state to *Idle*. If the local system receives a NOTIFICATION message, it changes its state to *Idle*. If the *KeepAlive* timer expires, the local system sends a KEEPALIVE message and restarts its *KeepAlive* timer. If a disconnect notification is

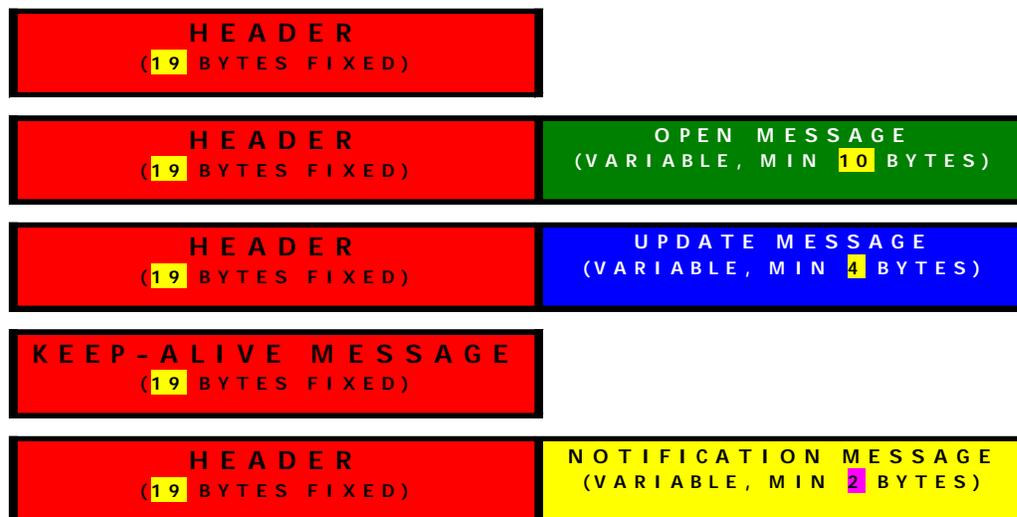
received from TCP, the local system changes its state to *Idle*. In response to the *Stop* event (initiated by either system or operator) the local system sends a NOTIFICATION message with an error code *Cease* and changes its state to *Idle*. The *Start* event is ignored in the *OpenConfirm* state. In response to any other event the local system sends a NOTIFICATION message with an error code *Finite State Machine Error* and changes its state to *Idle*. Whenever BGP changes its state from *OpenConfirm* to *Idle*, it closes the BGP (and TCP) connection and releases all resources associated with that connection (Rekhter, 1995).

Finally, in the *Established* state, BGP can exchange UPDATE, NOTIFICATION, and KEEPALIVE messages with its peer. If the local system receives an UPDATE or KEEPALIVE message, it restarts its *Hold Timer*, if the negotiated *Hold Time* value is non-zero. If the local system receives a NOTIFICATION message, it changes its state to *Idle*. If the local system receives an UPDATE message and the UPDATE message error-handling procedure detects an error, the local system sends a NOTIFICATION message and changes its state to *Idle*. If a TCP disconnect notification is received, the local system changes its state to *Idle*. If the *Hold Timer* expires, the local system sends a NOTIFICATION message with an error code *Hold Timer Expired* and changes its state to *Idle*. If the *KeepAlive* timer expires, the local system sends a KEEPALIVE message and restarts its *KeepAlive* timer. Each time the local system sends a KEEPALIVE or UPDATE message, it restarts its *KeepAlive* timer, unless the negotiated *Hold Time* value is zero. In response to the *Stop* event (initiated by either system or operator), the local system sends a NOTIFICATION message with an error code *Cease* and changes its state to *Idle*. The *Start* event is ignored in the *Established* state. In response to any other event,

the local system sends NOTIFICATION message with an error code *Finite State Machine Error* and changes its state to *Idle*. Whenever BGP changes its state from *Established* to *Idle*, it closes the BGP (and TCP) connection, releases all resources associated with that connection, and deletes all routes derived from that connection. For more detail on the BGP FSM and the associated timers and events, see RFC's 1771 and 1772 (Rekhter, 1995).

### E. BGP MESSAGE FORMATS

All BGP messages are sent over a TCP connection. A message is processed only after it is received in its entirety. The maximum message size is 4096 octets. All implementations are required to support this maximum message size. There are four BGP message types. These are the OPEN message, the UPDATE message, the KEEPALIVE message and the NOTIFICATION message (Figure 6).



**Figure 6: BGP Header and Four Message Types**

Regardless of type, each message contains a fixed-size header of 19 bytes. Therefore, even though there may not be a data portion following the header, such as in the case of the KEEPALIVE message, the minimum BGP message length will still be at least 19 octets.

### BGP Header Format

The BGP header has three different fields (Figure 7).

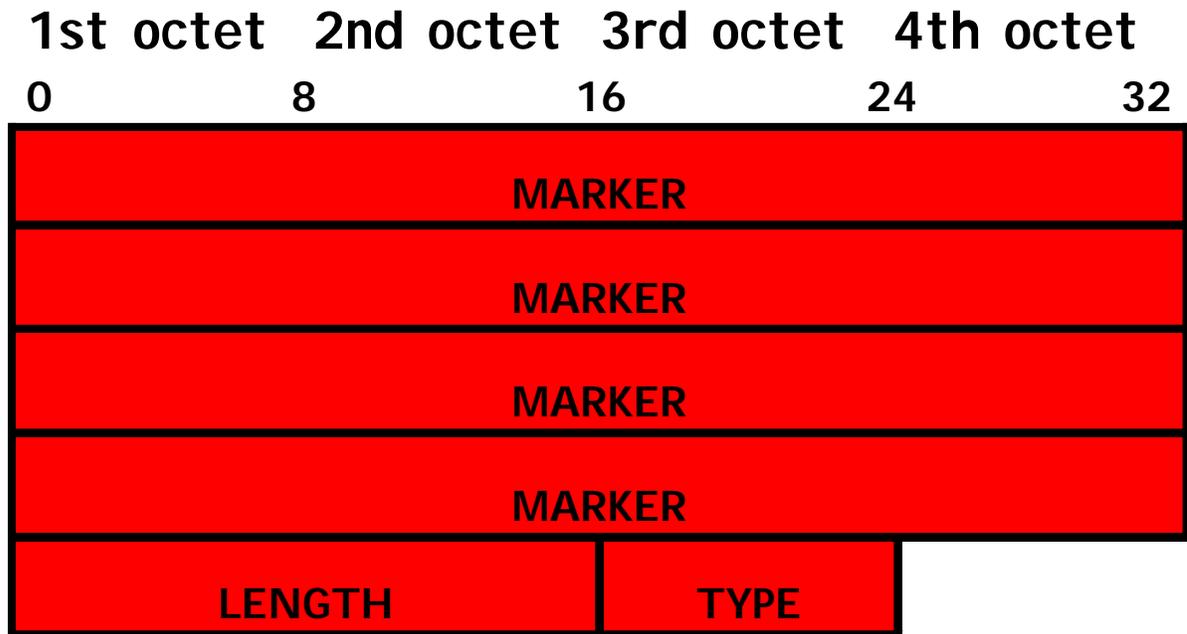


Figure 7: BGP Fixed-Size Header

The 16-octet *Marker* field is what is known as a field that contains ‘predicted’ values. In other words, the field contains a value that the receiver of the message can predict. It exchanges coordination and authentication information (it actually sets a flag that informs that authentication information will be sent as an *Optional Parameter* in an OPEN message to follow) between the BGP message sender and receiver. For example,

the value of the *Marker* can be predicted by some computation specified as part of an authentication mechanism used. The *Marker* can also be used to detect loss of synchronization between a pair of BGP peers, and to authenticate incoming BGP messages. If the *Type* of the message is OPEN, or if the OPEN message carries no authentication information, then all 16 octets of the *Marker* must be all ones.

The *Length* field is a two-octet unsigned integer indicating the total length of the message, including the header, in octets. Thus, it allows one to locate in the transport-level stream the beginning of the next message. The value of the *Length* field must always be at least 19 and no greater than 4096, and may be further constrained, depending on the message type. No "padding" of extra data after the message is allowed, so the *Length* field must have the smallest value required given the rest of the message (Rekhter, 1995).

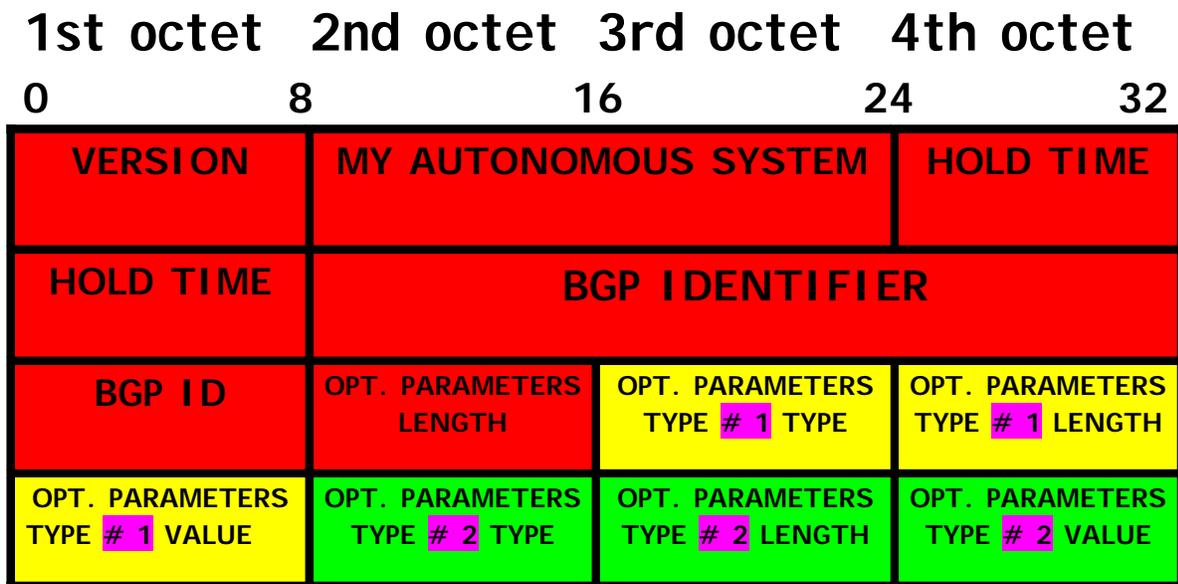
The *Type* field is a one-octet unsigned integer that indicates the type code of the message. There are four values of type codes that equate to the four types of BGP messages. They are:

- 1 – OPEN
- 2 – UPDATE
- 3 – NOTIFICATION
- 4 – KEEPALIVE.

### **OPEN Message Format**

The OPEN message is the first message sent between two BGP speakers after a TCP connection is established. If the OPEN message is acceptable, a KEEPALIVE

message confirming the OPEN message is sent back. Once the OPEN message is confirmed, then follow on UPDATE, KEEPALIVE, and NOTIFICATION messages may be exchanged. In addition to the fixed-size BGP header, the OPEN message contains the seven mandatory fields and several optional fields. The minimum length of the OPEN message is 29 octets, including the message header. The seven mandatory fields (ten octets total) are highlighted in red in Figure 8 below.



**Figure 8: OPEN Message Format**

This first field, *Version*, is a one-octet unsigned integer indicating the protocol version number of the message. The current BGP version number is four. The second field, *My Autonomous System*, is a two-octet unsigned integer indicating the AS number of the sender. The *Hold Time*, is a two-octet unsigned integer that indicates the number of seconds that the sender proposes for the value of the *Hold Timer* (as previously discussed in Section III.D, in relation to the BGP FSM). Upon receipt of an OPEN

message, a BGP speaker must calculate the value of the *Hold Timer* by using the smaller of its configured *Hold Time* and the *Hold Time* received in the OPEN message. The *Hold Time* must be either zero or at least three seconds. This is important because an implementation may reject connections on the basis of the *Hold Time*. The calculated value indicates the maximum number of seconds that may elapse between the receipt of successive KEEPALIVE, and/or UPDATE messages by the sender.

The *BGP Identifier* is a four-octet field indicating the BGP Identifier of the sender. A given BGP speaker sets the value of its *BGP Identifier* to an IP address assigned to that BGP speaker. The value of the BGP Identifier is determined on startup and is the same for every local interface and every BGP peer.

The *Optional Parameters Length* is one-octet field indicating the total length of the *Optional Parameters* field in octets. If the value of this field is zero, no *Optional Parameters* are present. The *Optional Parameters* field may contain a list of optional parameters, where each parameter is encoded as a three-octet triplet of *Parameter Type*, *Parameter Length*, and *Parameter Value*. Note the yellow and green highlighted triplets in Figure 8.

The *Parameter Type* is an optional one-octet field that identifies individual parameters. *Parameter Length* is an optional one-octet field that contains the length of the *Parameter Value* field in octets. *Parameter Value* is a variable length field that is interpreted according to the value of the *Parameter Type* field (Rekhter, 1995).

There are several types of *Optional Parameters*, but only one of the most important and frequently used types will be discussed here. See RFC 1771 for additional details and parameters. This important type is the *Authentication Information (Parameter*

*Type = 1*), which may be used to authenticate a BGP peer, has a *Parameter Value* field containing a one-octet *Authentication Code* followed by variable length *Authentication Data*. This *Authentication Code* is a one-octet unsigned integer indicating the authentication mechanism being used. Whenever an authentication mechanism is specified for use within BGP, three things must be included in the specification:

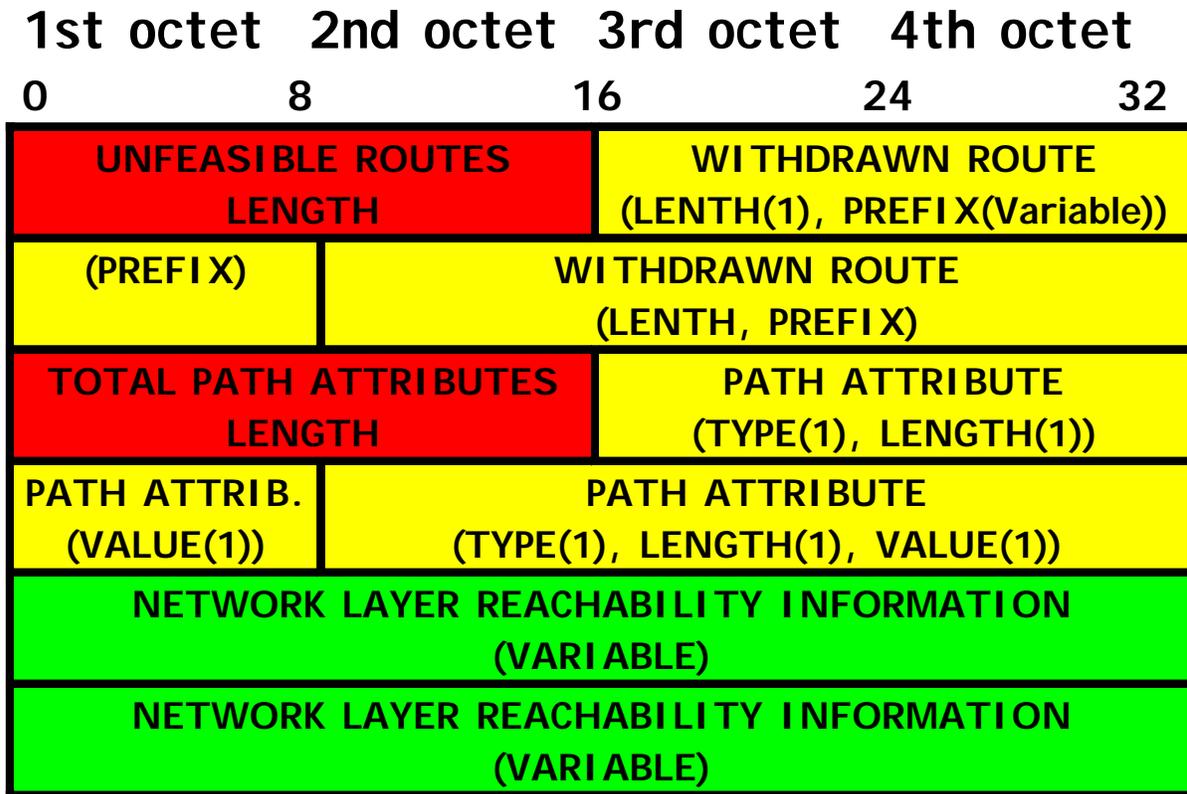
- the value of the Authentication Code which indicates use of the mechanism,
- the form and meaning of the Authentication Data, and
- the algorithm for computing values of Marker fields.

Note that a separate authentication mechanism may be used in establishing the TCP connection. The form and meaning of the variable-length *Authentication Data* field depends on the *Authentication Code*.

### **UPDATE Message Format**

UPDATE messages are used to transfer routing information between BGP peers. This message is really the heart of one of BGP's greatest contributions to the large-scale internetworking, since it simplifies and reduces the amount of information flow between ASs. The information in the UPDATE packet can be used to construct a graph describing the relationships of the various ASs. By applying rules and some basic metrics to be discussed in Sections III.F. and III.G. to follow, routing information loops and some other anomalies may be detected and removed from inter-AS routing. An UPDATE message is used to advertise a single feasible route to a peer, or to withdraw multiple unfeasible routes from service. In fact, one message may simultaneously advertise a feasible route and withdraw multiple unfeasible routes from service. The UPDATE message always

includes the fixed-size, 19-octet BGP header (see Figures 6 and 7 above), and can optionally include other fields (see Figure 9 below). Thus, the minimum length of the UPDATE message is 23 octets; 19 octets for the fixed header, plus two mandatory octets for the *Unfeasible Routes Length*, and two more octets for the *Total Path Attribute Length* (in this case the value of both the *Unfeasible Routes Length* and *Total Path Attribute Length* is zero).



**Figure 9: UPDATE Message Format**

The two-octet *Unfeasible Routes Length* indicates the total length of the *Withdrawn Routes* field in octets. Note that if a value of zero is used, it indicates that no

routes are being withdrawn from service, and that the *Withdrawn Routes* field is not present in this UPDATE message.

*Withdrawn Routes* is a variable length field (Figure 7 shows two consecutive three-octet fields as an example) that contains a list of IP address prefixes for the routes that are being withdrawn from service. Each IP address prefix is encoded in the form of the form  $\langle \text{Length}, \text{Prefix} \rangle$ . The *Length* field indicates the length in bits of the IP address prefix. The *Prefix* field contains IP address prefixes followed by enough trailing bits to make the end of the field fall on an octet boundary.

*Total Path Attribute Length* is a two-octet field indicating the total length of the *Path Attributes* field in octets. Once again, a value of zero indicates that no Network Layer Reachability Information field is present in this UPDATE message.

*Path Attributes* is a variable length sequence of path attributes present in every UPDATE Message. Each path attribute is a variable length triplet of the form  $\langle \text{Attribute Type}, \text{Attribute Length}, \text{Attribute Value} \rangle$ . There are seven main *Attribute Types* and their formats and lengths vary. These types are listed by their respective *Attribute Type* code:

- 1 - ORIGIN
- 2 - AS\_PATH
- 3 - NEXT\_HOP
- 4 - MULTI\_EXIT\_DISC
- 5 - LOCAL\_PREF
- 6 - ATOMIC\_AGGREGATE
- 7 - AGGREGATOR

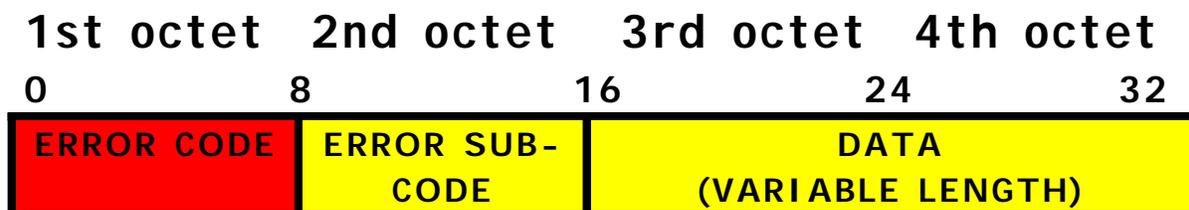
*Attribute Length* indicates the length of the *Attribute Values* to follow. *Attribute Values* fall into four separate categories, which basically limit and control whether or not a particular attribute is passed along to other BGP peers:

1. Well-known mandatory.
2. Well-known discretionary.
3. Optional transitive.
4. Optional non-transitive.

Some specifics of a few of the *Attribute Types* (Codes 2, 3, 4 and 5) will be covered in the following sections as they directly relate to some performance issues, but no specifics on the *Attribute Values* will be covered here. For this level of detail on every *Attribute Type* and *Value*, the interested reader is referred to RFC 1771 (Rekhter, 1995).

### **NOTIFICATION Message Format**

A NOTIFICATION message is sent when an error condition is detected. The BGP connection is closed immediately after sending it. In addition to the fixed-size BGP header, the NOTIFICATION message contains three other fields (see Figure 10 below);



**Figure 10: NOTIFICATION Message Format**

the Error Code, Error *Subcode* and the *Data* field. The one-octet *Error Code* indicates the type of NOTIFICATION Message. There are six basic *Error Codes*:

- 1 - Message Header Error
- 2 - OPEN Message Error
- 3 - UPDATE Message Error
- 4 - Hold Timer Expired
- 5 - Finite State Machine Error
- 6 - Cease (Interruption by system or administrator)

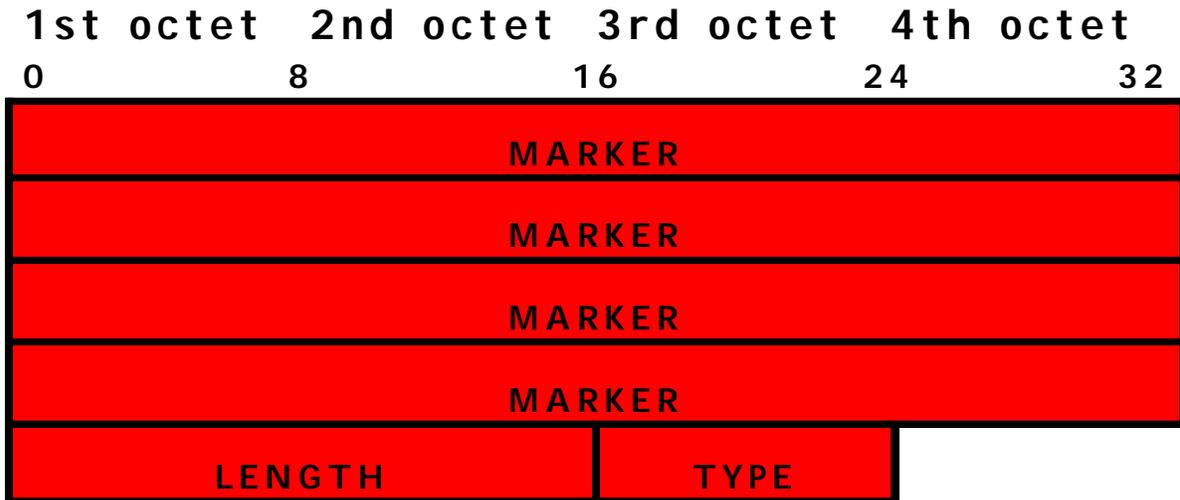
The one-octet *Error Subcode* provides further amplification about the nature of the reported error. Each *Error Code* has different *Error Subcodes* associated with it and in some cases may contain multiple sub-codes (for details on the numerous types of codes, see RFC 1771). If no appropriate *Error Subcode* is defined for a particular error, then a zero (unspecific) value is used for this field.

The variable-lengthed *Data* field is used to diagnose the reason for the NOTIFICATION and therefore, its contents are dependent upon the first two fields. The minimum length of the NOTIFICATION message is 21 octets (including the BGP header).

### **KEEPALIVE Message Format**

BGP does not use TCP's keep-alive mechanism to determine if peers are reachable. Instead, KEEPALIVE messages are exchanged between peers often enough

as not to cause the *Hold Timer* to expire. The KEEPALIVE message consists of only the message header and has a length of 19 octets (see Figure 11 below).



**Figure 11: KEEPALIVE Message Format**

RFC 1771 recommends that a reasonable maximum time between KEEPALIVE messages should be one third of the *Hold Time* interval. The RFC also warns that KEEPALIVE messages must not be sent more frequently than one per second. An implementation may adjust the rate at which it sends KEEPALIVE messages as a function of the *Hold Time* interval. Therefore, if the negotiated *Hold Time* interval is zero, then periodic KEEPALIVE messages must not be sent.

## F. BGP METRICS AND PATH ATTRIBUTES

BGP speakers use several metrics and path attributes to control traffic flow and influence decisions in its routing algorithm (BGP's algorithm is introduced in Section

III.G to follow). There are five metrics that are of primary interest for the focus of this paper, since they have the greatest influence on the BGP algorithm, the method by which BGP selects a path to route its traffic. The first four were introduced earlier in Section III.E. during the coverage of the UPDATE message *Path Attributes*. These are the *Attribute Type* codes 2, 3, 4 and 5, which correspond to the AS\_PATH, NEXT\_HOP, MULTI\_EXIT\_DISC (or MED), LOCAL\_PREF path attributes, respectively. The fourth metric is the WIEGHT metric, which is a configuration metric used by a locally BGP-configured router, and not passed on to any other router.

#### **AS\_PATH Path Attribute**

AS\_PATH is a mandatory attribute that is basically a routing list that identifies the ASs through which routing information carried in each BGP UPDATE message has actually passed. The components of this routing list can be AS\_SETs or AS\_SEQUENCES.

When a BGP speaker propagates a route (receives and retransmits an UPDATE message) which it has learned from another BGP speaker's UPDATE message, it, in turn, modifies the route's AS\_PATH attribute depending on the location of the BGP speaker to which the route will be sent. In other words, if the BGP speaker advertises the route to another BGP speaker located in its own AS, the advertising speaker doesn't modify the AS\_PATH attribute associated with the route. However, if a given BGP speaker advertises the route to a BGP speaker located in a neighboring AS, then the advertising speaker updates the AS\_PATH attribute.

When a BGP speaker is the originator of a route then the originating speaker includes its own AS number in the AS\_PATH attribute of all UPDATE messages sent to BGP speakers located in neighboring ASs (In this case, the AS number of the originating speaker's autonomous system will be the only entry in the AS\_PATH attribute). Also, the originating speaker sends UPDATE messages with an empty AS\_PATH attribute in to all internal BGP speakers located within in its own AS.

### ***NEXT\_HOP Path Attribute***

The NEXT\_HOP path attribute defines the IP address of the border router that should be used as the next hop to the destinations listed in the UPDATE message. If a border router belongs to the same AS as its peer, then the peer is an internal border router. Otherwise, it is an external border router. A BGP speaker can advertise any internal border router as the next hop provided that the interface associated with the IP address of this border router (as specified in the NEXT\_HOP path attribute) shares a common subnet with both the local and remote BGP speakers. A BGP speaker can advertise any external border router as the next hop, provided that the IP address of this border router was learned from one of the BGP speaker's peers, and the interface associated with the IP address of this border router (as specified in the NEXT\_HOP path attribute) shares a common subnet with the local and remote BGP speakers.

In order to prevent routing loops a BGP speaker has some very important rules. It must never advertise an address of a peer to that peer as a NEXT\_HOP, for a route that the speaker is originating. A BGP speaker must never install a route with itself as the next hop. When a BGP speaker advertises the route to a BGP speaker located in its own

AS, the advertising speaker shall not modify the NEXT\_HOP attribute associated with the route. When a BGP speaker receives the route via an internal link, it may forward packets to the NEXT\_HOP address if the address contained in the attribute is on a common subnet with the local and remote BGP speakers.

### **MULTI\_EXIT\_DISC (MED) *Path Attribute***

When two ASs have multiple links with each other, BGP uses the four-octet Multi-Exit Discriminator attribute (MULTI\_EXIT\_DISC or MED) in its UPDATE message to inform the other AS of the preferred entrance point. The MED is an ‘inbound’ metric that assists in a form of route mapping. It is simply a hint to external neighbors about the recommended path into an AS when there are multiple entry points into that AS. The external neighbor still remembers the other paths with the higher MED values in case the pathway through the lower MED link becomes unavailable. All other factors being equal, a lower MED value is preferred over a higher MED value. The default value of the MED attribute is zero.

Unlike the LOCAL\_PREF (discussed below), the MED attribute is exchanged between ASs, but a MED attribute that comes into an AS does not leave the AS. The MED attribute is never propagated to other BGP speakers in neighboring AS's. When an update enters the AS with a certain MED value, that value is used for decision making within that AS. When the second or neighboring AS does advertise the networks from the originating AS, the MED value is set back to zero before leaving the second AS.

### **LOCAL\_PREF Path Attribute**

LOCAL\_PREF is a discretionary attribute that is included in all UPDATE messages that a BGP speaker sends to the other BGP speakers located in its own AS, or iBGP. THE LOCAL\_PREF attribute is an ‘outbound’ metric used within an AS between inA BGP speaker shall calculate the degree of preference for each external route and include the degree of preference when advertising a route to its internal peers. The higher degree of preference should be preferred. A BGP speaker shall use the degree of preference learned via LOCAL\_PREF in its decision process (see Section III.G.). A BGP speaker shall not include this attribute in UPDATE messages that it sends to BGP speakers located in a neighboring autonomous system. If it is contained in an UPDATE message that is received from a BGP speaker which is not located in the same autonomous system as the receiving speaker, then this attribute shall be ignored by the receiving speaker.

### **WIEGHT Metric**

The WEIGHT metric allows a ‘weight’ to be assigned to all routes originating from the specified neighbor or group of neighbors. This router only uses the metric. If the same network is learned from two different neighbors, the neighbor with the highest weight assigned will be the one selected to receive the packet or packets. The BGP configured router will not pass the WEIGHT metric on to any other router.

## G. BGP ALGORITHM

BGP uses a ten-step algorithm (see Figure 12 below) in order to reach a decision about the one, and only one, path it will select for to reach a destination to a given AS. Its very first step is to check to see if the next hop (as per the *NEXT+HOP Path Attribute*) that is advertised is even reachable. If so, then it continues through the remaining nine steps (two through nine in ascending order), and only applying them as applicable.

### **BGP selects a path based on the following priorities:**

- 1) If NEXT\_HOP is unreachable, do not use that update.
- 2) Prefer the path with the largest WEIGHT.
- 3) If no WEIGHT or the same WEIGHT, select the largest LOCAL\_PREF.
- 4) If the same LOCAL\_PREF, prefer the path (if any) that was originated by BGP on this router.
- 5) If no route was originated, prefer the shorter AS\_PATH.
- 6) If all paths are the same length, prefer the lowest ORIGIN code: (iBGP < eBGP < Incomplete).
- 7) If origin codes are the same, prefer the path with the lowest MULTI\_EXIT\_DISC (MED).
- 8) If paths have the same MED, prefer the External path over Internal.
- 9) If paths are still equal, prefer the path with the closest IGP neighbor (lowest cost).
- 10) Prefer the path with the lowest BGP router ID.

**Figure 12: The BGP Algorithm**

Once one of the conditions (steps) is satisfied it does not continue through the rest of the steps of the algorithm.

Note that the WEIGHT metric, the LOCAL\_PREF attribute and the AS\_PATH attribute are the most influential policy parameters on the path selection process. For example, when the WEIGHT metric is used, it will dominate all decisions for path selection. It is the second step of the algorithm, but it is the first real ‘configurable’ or policy enforcing metric, since the answer to the NEXT\_HOP condition is either an affirmation or negation. This strong influence of the WEIGHT metric makes sense since the WEIGHT metric is an outbound metric used by only a given router and does not influence other routers or ASs.

Also, notice that the MED attribute doesn’t play a role in the decision process until the seventh step. This helps to create a protective buffer to control incoming traffic in a given AS, but still allows flexibility for adjustments if preferred pathways become unavailable.

The last step is basically a catch-all condition that one might assume would only be reached in the case of a loosely configured BGP speaking system, but that is not always the case. If it comes down to choosing a path based on the lowest value of a BGP router ID, it simply means that all other conditions were equal or not applicable.

## IV. PROTOCOL ANALYSIS

### A. KEY FEATURES

This section summarizes the key features of the BGP-4 protocol and explains some of the advantages of its use as compared to its predecessor EGP. Since BGP is an inter-AS routing protocol, it is designed to be used between multiple ASs. BGP makes an assumption that routing within an AS is done by an intra-AS routing protocol or IGP. BGP does not make any assumptions about the different IGPs employed by the various ASs. This is a very advantageous feature of BGP, since it does not require all ASs to run the same IGP. It imposes no constraints on the underlying Internet topology. This further defines BGP as a ‘true’ inter-AS routing protocol and separates it from the former EGP, which imposed certain restrictions and limitations on various ASs’ IGPs.

BGP is a self-contained protocol and not only adaptable to the coexistence of other IGP’s but it also tolerates other inter-AS routing protocols, if a neighboring AS so desires to implement one. That is, BGP not only specifies how routing information is exchanged between BGP speakers within an AS, but also between BGP speakers in different ASs. For example, to allow graceful coexistence with EGP and OSPF, BGP provides support for carrying both EGP and OSPF derived exterior routes. BGP also accepts statically defined exterior routes or routes derived by other IGP information.

As discussed previously, the information exchanged via BGP is sufficient to construct a graph of AS connectivity from which routing loops may be pruned and routing policy decisions at the AS level may be enforced. Although routing loops can still occur, prior

to the use of BGP, routing loops were much more prevalent with EGP. There was simply not an existing method of addressing the pruning issue since the hop-to-hop paradigm was from router to router in a link-state manner without the benefit of the knowledge of an entire AS path (i.e. AS\_PATH, AS\_SEQUENCE, AS\_SET).

This further explains why the AS\_PATH attribute is so beneficial. As AS reachability information traverses the Internet, this information is augmented by the list of AS that have been traversed thus far, forming the AS-PATH. The AS-PATH allows straightforward suppression of the looping of routing information. By having the full AS path information available to the its algorithm, BGP can not only prune routing loops, but it can make smarter decisions about choosing between overlapping paths. The term overlapping in this context means routes that have almost all hops that are similar in their AS\_SET or AS\_SEQUENCE with the exception of one or two hops. For example, there may be several overlapping paths to a given network, but BGP will only choose one. It will choose the best path, which is normally the shortest path when no other policy enforcement exists. So if the route lengths happen to reach a tie, the BGP algorithm works out the difference by comparing the assigned metrics and path attributes and breaks the tie.

In addition, the AS\_PATH attribute serves as a powerful and versatile mechanism for policy-based routing. The AS\_SET and AS\_SEQUENCE options of AS\_PATH allows generated aggregate routes to carry path information from the more specific routes used to generate the aggregate. The BGP algorithm cannot be classified as either a pure distance vector, or a pure link state. Carrying a complete AS path in the AS-PATH attribute allows reconstruction of large portions of the overall topology. That makes it

similar to the link state algorithms. Exchanging only the currently used routes between the peers makes it similar to the distance vector algorithms.

Another key feature that separates BGP from EGP is the notion of *Path Attributes* as they relate to the aggregation of network layer reachability information (NLRI). This concept provides BGP with awesome flexibility and expandability. *Path Attributes* are partitioned into well-known and optional. The provision for optional attributes allows experimentation that may involve a group of BGP routers without affecting the rest of the Internet (Rekhter. 1995). For instance, new optional attributes can be added to the protocol in much the same fashion as new options are added to the Telnet protocol as seen fit.

A major advantageous feature of BGP is its use of the UPDATE message. To conserve bandwidth and processing power, BGP uses incremental updates, where after the initial exchange of complete routing information, a pair of BGP routers exchanges only changes (deltas) to that information. This technique of incremental updates requires a reliable transport between a pair of BGP routers, hence BGP's. Prior to BGP, EGP sent complete routing table updates in order to keep routing information. The benefits of this improvement are quite obvious, since bandwidth and processing power are adversely affected otherwise.

In addition to incremental updates, BGP also added the concept of route aggregation so that information about groups of networks may be represented as a single entity. This was not a feature of EGP.

## **B. PERFORMANCE**

This section will address the more engineering performance related issues. RFC 1774, written in 1995, addresses in detail some issues such as how much link bandwidth, router CPU cycles and router memory requirements the BGP protocol may consume under normal conditions. All three of these issues directly relate to the scalability of BGP. RFC 1774 also addresses BGP's performance limits and introduces the importance on overall Internet stability on BGP's performance. For the purpose of this paper, some of the key points and predictions of the RFC will be summarized and then compared with what has actually occurred with BGP in the four year since the protocols acceptance as a standard. Some of the reasons for the successful operational experience with BGP on the Internet will be addressed in what follows.

Recall some of the scaling issues that were discussed in the Introduction of this paper; mainly the size of Internet routing tables growing to immensely and almost unmanageable sizes. A less obvious and quite interesting feature of BGP is that it does not require all the routers within an AS to participate in the BGP protocol. Only the border routers that provide connectivity between the local ASs and its adjacent ASs participate in BGP. This feature is a minor point but it does directly address some of the scaling issues that were introduced in Section I, by limiting the amount of BGP participants to only those that are necessary.

### **Link Bandwidth**

Both link bandwidth and CPU utilization are important parameters for almost any router since it is a traffic convergence area and or potential bottleneck region. However,

these parameters take on even greater relevance at the gateway routers or ASs. RFC 1774 gives an equation to estimate the bandwidth consumption of initial information exchanges between two BGP speakers:

Immediately after the initial BGP connection setup, the peers exchange complete set of routing information. If we denote the total number of routes in the Internet by N, the mean AS distance of the Internet by M (distance at the level of an autonomous system, expressed in terms of the number of autonomous systems), the total number of autonomous systems in the Internet by A, and assume that the networks are uniformly distributed among the autonomous systems, then the worst case amount of bandwidth consumed during the initial exchange between a pair of BGP speakers is

$$B = 3.5(N + M * A)$$

Figure 13 illustrates typical amount of bandwidth consumed during the initial

<b># NLRI (N)</b>	<b>Mean AS Distance (M)</b>	<b># AS's (A)</b>	<b>Bandwidth (B)</b>
10,000	15	300	49,000 bytes
20,000	8	400	86,000 bytes
40,000*	15	400	172,000 bytes
100,000	20	3,000	520,000 bytes

\* the actual "size" of the Internet at the time of RFC 1774's publication in 1995

**Figure 13: BGP Bandwidth Consumption**

exchange between a pair of BGP speakers based on the above assumptions (ignoring bandwidth consumed by the BGP Header): Since RFC 1774 was written in 1995, the number of routes (or prefixes) in the current Internet have actually reached about 63000. See Figure 14 below, which graphs the actual number of the global Internet's network prefixes on a BGP table located in a MAE in Australia since 1994 to the present (22 March 1999). Using the suggested equation above, and using values  $N=63000$  prefixes (from the current value of today's Internet from Figure 14), and estimations of  $M=17$  average hop distance and  $A=2000$ , we get a BGP bandwidth consumption of about 339,500 bytes for today's Internet.

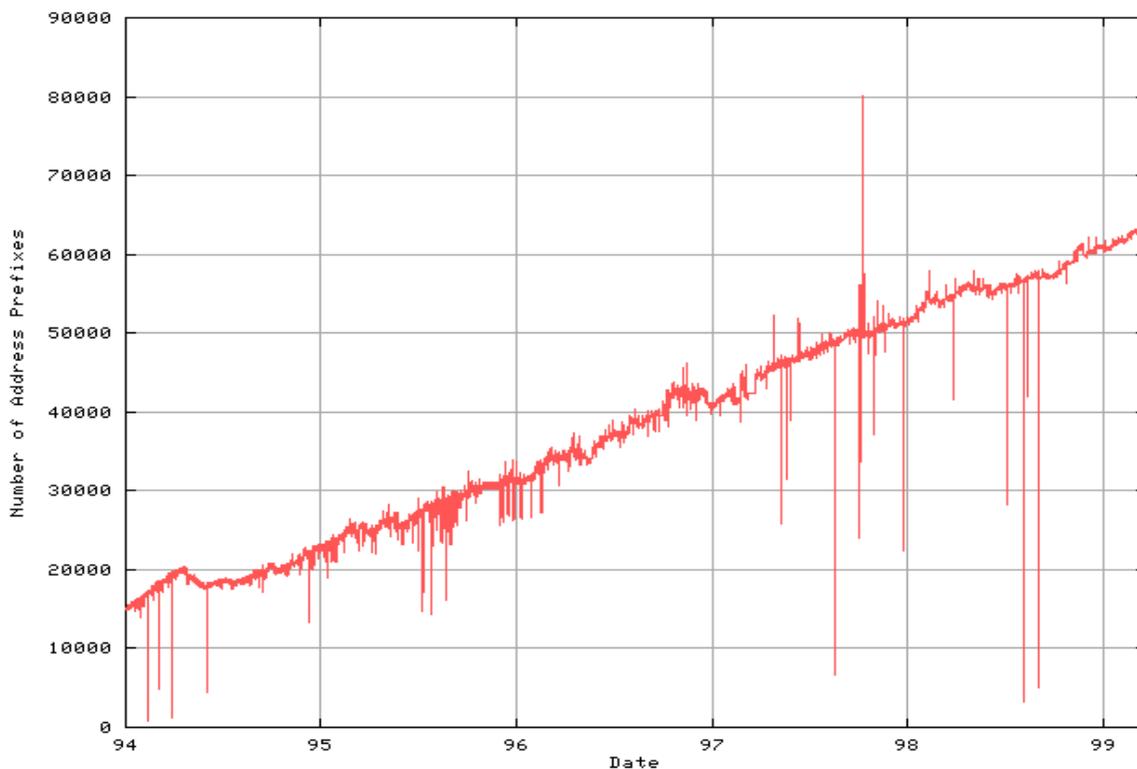


Figure 14: Internet BGP Prefixes from 1994-Present

(Source: Telstra Internet, <http://www.telstra.net/ops/bgptable.html>)

Checking this value with BGP tables in USA's MAE West, located in San Jose, CA, we see we arrive at a relatively good estimation. Current estimations of bandwidth consumption for initial exchange between two BGP Speakers in MAE West is approximately 400Kbytes.

Note that most of the bandwidth is consumed by the exchange of the Network Layer Reachability Information (NLRI). BGP-4 was created specifically to reduce the amount of NLRI entries carried and exchanged by border routers. BGP-4, along with CIDR, introduced the concept of supernetting, as discussed in Section II. Due to the advantages of advertising a few large aggregate blocks instead of many smaller class-based individual networks, it is difficult to estimate the actual reduction in bandwidth and processing that BGP-4 has provided over BGP-3, which did not have this feature. If we simply enumerate all aggregate blocks into their individual class-based networks, we would not take into account "dead" space that has been reserved for future expansion. The best metric for determining the success of BGP-4's aggregation is to sample the number NLRI entries in the globally connected Internet today and compare it to projected growth rates before BGP-4 was deployed (Traina, 1995).

In January of 1994, router carrying a full routing load for the globally connected Internet had approximately 19,000 network entries (this number is not exact due to local policy variations). The BGP deployment working group estimated that the growth rate at that time was over 1000 new networks per month and increasing. Since the widespread deployment of BGP-4, the growth rate has dropped significantly and a sample done at the end of November 1994 showed approximately 21,000 entries present, as opposed to the

expected 30,000 (Traina, 1995). Likewise, the number of prefixes that were expected by January 1999 were approximately 100,000, and we are presently at only two-thirds of that value.

Where the BGP-4 to BGP-3 comparisons are more difficult, the BGP-4 to EGP comparison is quite simple. Without any subnetting, aggregation, supernetting and by simply advertising every single host's network ID to the Internet gateway, we would be in serious trouble. The current estimation of Internet hosts is about 63 million. Using the previous equation, we can roughly estimate that it may have required a link bandwidth consumption of 225Mbytes for initial EGP routing table exchanges!

### **Router CPU Cycles**

CPU cycles consumed by BGP depends only on the stability of the Internet (Traina, 1995). If the Internet is stable, then the only link bandwidth and router CPU cycles consumed by BGP are due to the exchange of the BGP KEEPALIVE messages. The KEEPALIVE messages are exchanged only between peers. The suggested frequency of the exchange is 30 seconds. The KEEPALIVE messages are quite short (19 octets), and require virtually no processing. Therefore, the bandwidth consumed by the KEEPALIVE messages is about 5 bits/sec (Traina, 1995).

RFC 1774 states that operational experience confirms that the overhead (in terms of bandwidth and CPU) associated with the KEEPALIVE messages should be viewed as negligible (Triana, 1995). If the Internet is unstable, then only the changes to the reachability information (that are caused by the instabilities) are passed between routers

(via the UPDATE messages). RFC 1774 suggest an equation for worst case bandwidth due to routing changes:

“If we denote the number of routing changes per second by  $C$ , then in the worst case the amount of bandwidth consumed by the BGP can be expressed as  $O(C * M)$ . The greatest overhead per UPDATE message occurs when each UPDATE message contains only a single network. It should be pointed out that in practice routing changes exhibit strong locality with respect to the AS path. That is routes that change are likely to have common AS path. In this case multiple networks can be grouped into a single UPDATE message, thus significantly reducing the amount of bandwidth required.”

Since in the steady state the link bandwidth and router CPU cycles consumed by the BGP protocol are dependent only on the stability of the Internet, but are completely independent on the number of networks that compose the Internet, it follows that BGP should have no scaling problems in the areas of link bandwidth and router CPU utilization, as the Internet grows, provided that the overall stability of the inter-AS connectivity (connectivity between ASs) of the Internet can be controlled (Triana, 1995).

### **Internet Stability**

It is important to point out, that regardless of BGP, one should not underestimate the significance of the stability in the Internet. Growth of the Internet has made the stability issue one of the most crucial ones. It is important to realize that BGP, by itself, does not introduce any instabilities in the Internet. It has been experienced over the past on the NSFNET and on today's Internet, that instabilities are largely due to the ill-behaved routing within the autonomous systems that compose the Internet. Therefore,

the way the engineers are addressing these issues is to come up with intra-AS routing schemes that exhibit reasonable stability. BGP does this by buffering the instabilities of one or several different intra-AS routing protocols that may co-exist in an a single AS.

## **Router Memory**

RFC 1774 also suggest an equation to quantify the worst case memory requirements for BGP (Triana, 1995);

Denote the total number of networks in the Internet by  $N$ , the mean AS distance of the Internet by  $M$  (distance at the level of an autonomous system, expressed in terms of the number of autonomous systems), the total number of autonomous systems in the Internet by  $A$ , and the total number of BGP speakers that a system is peering with by  $K$  (note that  $K$  will usually be dominated by the total number of the BGP speakers within a single autonomous system). Then the worst case memory requirements ( $MR$ ) can be expressed as  $MR = 3.5(N + M * A) * K$ .

Figure 15 below illustrates typical memory requirements of a router running BGP. It is assumed that each network is encoded as 4 bytes, each AS is encoded as 2 bytes, and each networks is reachable via some fraction of all of the peers (# BGP peers/per net).

When RFC 1774 was written in 1995, the NSFNET Backbone carried approximately 20,000 network or prefix advertisement entries. We are now, as mentioned previously around the 63000 prefix range on your average Internet Gateway router, which requires less than 1Mbytes of memory. These memory requirements are really not a requirement of BGP, but of router memory and is completely independent of

BGP. Using the current values for EGP, the memory requirement would be in excess of 100Mbytes!

# Networks	Mean AS Distance	# AS's	# BGP peers/per net	Memory Req
2,100	5	59	3	27,000
4,000	10	100	6	108,000
10,000	15	300	10	490,000
100,000	20	3,000	20	1,040,000

*(Source: RFC 1774, Traina, March 1995)*

**Figure 15: Router Memory Requirements (MR)**

The growing routing tables are really not a BGP issue but rather a lack of a form of hierarchy of the IP address format. IP has a flat address space (Traina, 1995). Because of the flat IP address space, any routing protocol must carry network numbers in its updates. Both CIDR and BGP-4 attempt to reduce this limitation of IP but they do not solve the problems inherent with inefficient assignment of future address blocks (as addressed in Section I). BGP's limits with respect to memory requirements are directly related to the underlying Internet Protocol (IP), and specifically the addressing scheme employed by IP. BGP would provide much better scaling in environments with more flexible addressing schemes. It should be pointed out that with only very minor additions BGP was extended to support hierarchies of ASs. Such hierarchies, combined with an addressing scheme that would allow more flexible address aggregation capabilities, can

be utilized by BGP-like protocols, thus providing practically unlimited scaling capabilities (Traina, 1995).

## V. CONCLUSION

BGP is not only well suited for the current Internet, it could be viewed as a necessity for the current Internet as well. Operational experience with EGP showed that it is highly inadequate for the current Internet even as early as 1995. EGP imposed topological restrictions that are unjustifiable from the technical point of view, and unenforceable from the practical point of view (Traina, 1995). The inability of EGP to efficiently handle information exchange between peers was a cause of severe routing instabilities in the operational Internet. Finally, information provided by BGP is well suitable for enforcing a variety of routing policies.

BGP was designed with simplicity in mind. The protocol contains only the functionality that is essential, while at the same time provides flexible mechanisms within the protocol itself that allow to expand its functionality. Since BGP was designed with flexibility and expandability in mind, it should be able to address new or evolving requirements with relative ease.

In summary, BGP is well suitable as an inter-AS routing protocol for the current Internet that is based on IP (RFC 791) and the "hop-by-hop" routing paradigm. Perhaps movements towards concepts such as hybrid router-switching technologies, Wave Division Multiplexed (WDM) or photonic routing, ATM multicasting, IPv6, real time applications and the Next Generation Internet (NGI) may play a role in changing the current routing paradigm. It is difficult to speculate whether BGP will be suitable for these other environments where internetworking may be done by other than IP protocols or where the routing paradigm will be different.

## LIST OF REFERENCES

Mills, D., "Exterior Gateway Protocol Formal Specification, RFC 904", BBN, April 1984.

Postel, J., "Transmission Control Protocol - DARPA Internet Program Protocol Specification, STD 7, RFC 793", DARPA, September 1981.

Rekhter, Y., "**A Border Gateway Protocol (BGP), RFC 1163**", T.J. Watson Research Center, IBM Corp., June 1990.

Rekhter, Y., "**A Border Gateway Protocol 3 (BGP-3), RFC 1267**", T.J. Watson Research Center, IBM Corp., October 1991.

Rekhter, Y. and Li, T., "**A Border Gateway Protocol 4 (BGP-4), RFC 1771**", T.J. Watson Research Center, IBM Corp, 1995.

Rekhter, Y., and P. Gross, "**Application of the Border Gateway Protocol in the Internet, RFC 1772**", T.J. Watson Research Center, IBM Corp., MCI, March 1995.

Rekhter, P. Traina, "**BGP-4 Protocol Analysis, RFC 1774**", Cisco Systems, March 1995.

Stevens, W. Richard, "**TCP/IP Illustrated Volume 1: The Protocols**", Addison-Wesley Publishing Company, 1994.

Stallings, William, "**Data and Computer Communications**", Prentice-Hall, 1997.

<http://joe.lindsay.net/bgp.html>

<http://www.cis.ohio-state.edu/htbin/rfc/rfc1771.html>

<http://www.cisco.com/warp/public/459/21.html>

[http://www.compatible.com/tech\\_support/tech\\_faq/supernetting.faq.html](http://www.compatible.com/tech_support/tech_faq/supernetting.faq.html)

<http://www.ibm.net.il/~hank/cidr.html>

<http://www.telstra.net/ops/bgptable.html>

<http://www.nw.com/>