

The Implementation of Error Checking and Optional Parameter Negotiation for BGP-4 in a Network Simulator

Naomi Ko
95301-1394

Communication Networks Laboratory
Simon Fraser University

2004 Sep 01



Presentation Overview

- Acknowledgements
- Introduction
- Background Knowledge
- Project Contribution
- Simulation Results
- Future Enhancements
- Conclusion
- References
- Questions

Acknowledgements



- Dr. Ljiljana Trajković
- Dr. Shahram Payandeh
- Tony D. Feng
- Nenad Lasković
- Jenny Koo and Mary Kwong
- Communication Networks Laboratory
- ... several people behind-the-scenes



Introduction

- Internet: a cluster of ASes
 - Within AS: Interior Gateway Protocols
 - Among ASes: Exterior Gateway Protocols
- Scalability and performance are key
- Interest in academic research community to investigate behaviour of networks
- Development of network simulation tools
 - SSFNet, OPNET, ns-2
- Improved accuracy of a network model better reflects actual behaviour

Background Knowledge

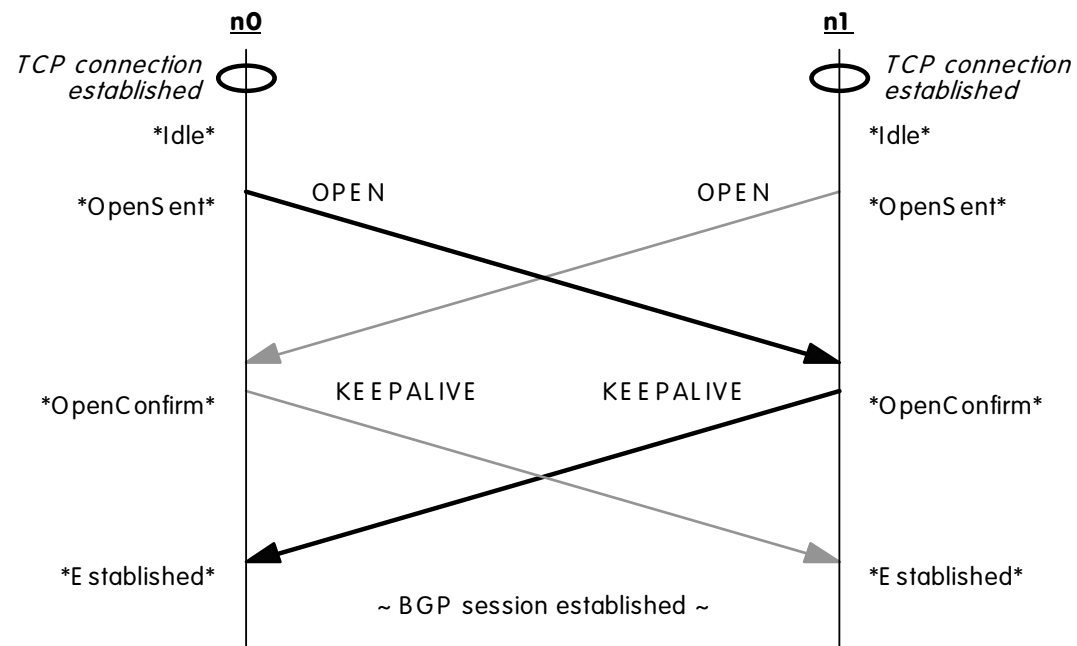


- Border Gateway Protocol
 - Exterior Gateway Protocol
 - BGP-4 is the *de facto* protocol
 - Operation using 4 message types
 - OPEN
 - UPDATE
 - NOTIFICATION
 - KEEPALIVE

Background Knowledge (cont'd)

- OPEN Process

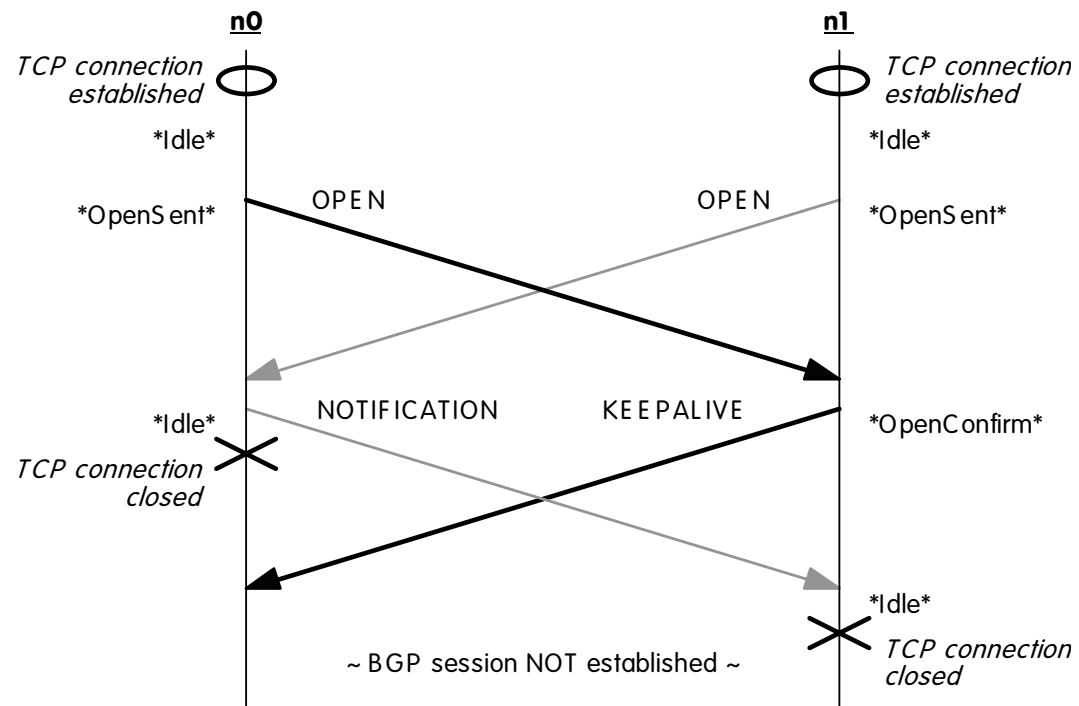
- 3-way handshaking procedure



Background Knowledge (cont'd)

● OPEN Process

- failed process: n0 deems parameter unacceptable





Background Knowledge (cont'd)

- BGP Simulation Models

- SSFNet model SSF.OS.BGP4
by Brian J. Premore [14]

- Java-based model

- ns-2 model ns-BGP
by Tony D. Feng [9]

- C++/OTcl model

- Ported over from SSFNet model

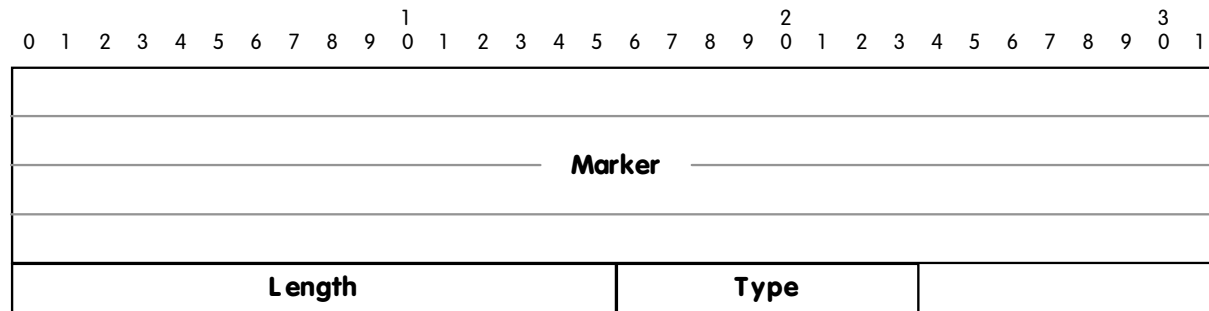
Project Contribution



- Original project plan out the window...
- Assumptions made to simplify SSFNet model during implementation
 - Carried over to ns-BGP
- Error checking
- OPEN message Optional Parameter
- Negotiation of optional parameters

Project Contribution: Error Checking

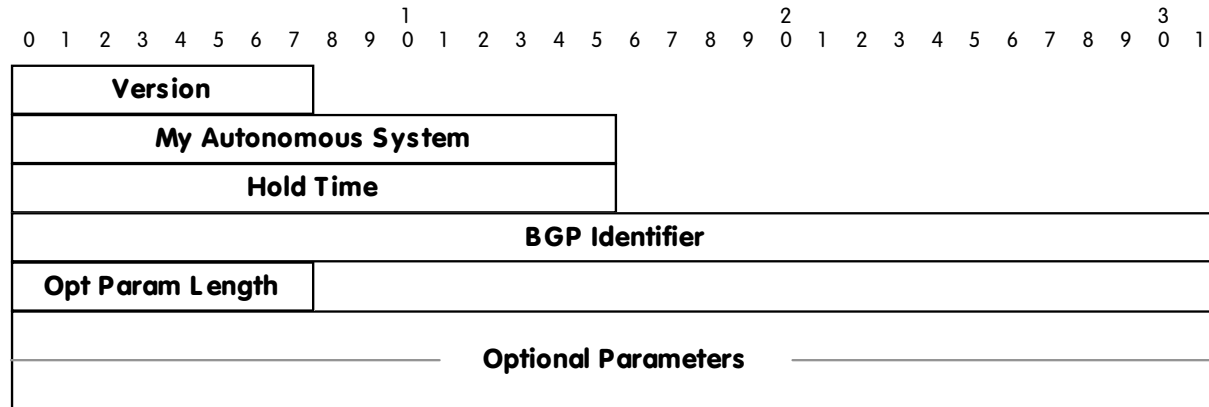
- BGP message header
 - Common 19-byte header



- Marker: used for Authentication Information
- Length: total length of BGP message
- Type: OPEN, UPDATE, NOTIFICATION, KEEPALIVE

Contribution: Error Checking (cont'd)

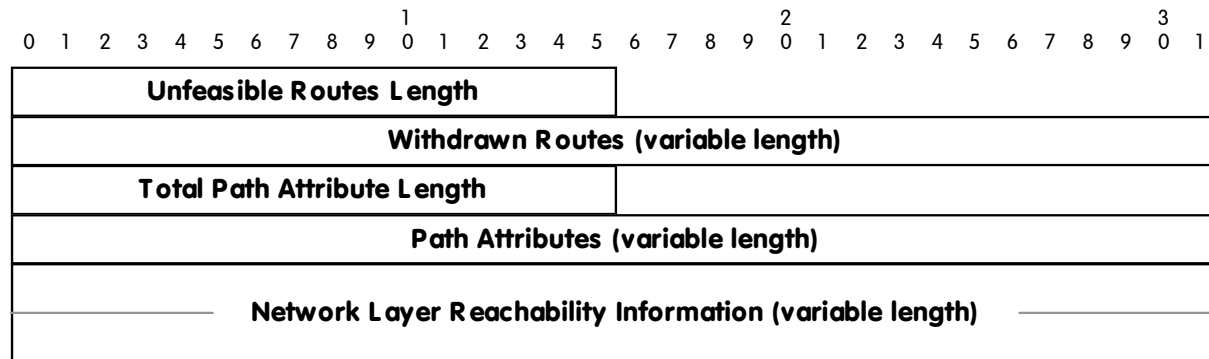
- OPEN message



- Handshaking information
 - Identification
 - Desired session parameters

Contribution: Error Checking (cont'd)

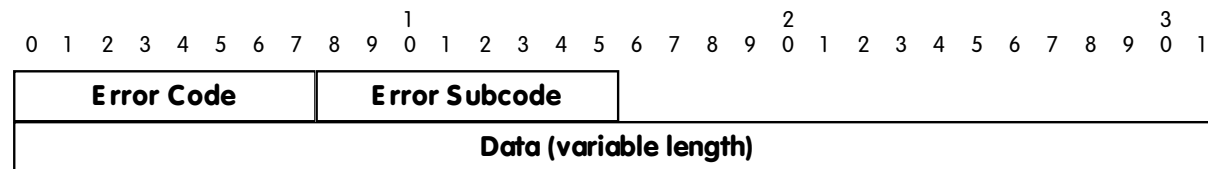
- UPDATE message



- Unfeasible Routes
- Path Attributes (for new routes)
- Network Layer Reachability Information (NLRI)

Contribution: Error Checking (cont'd)

- NOTIFICATION message



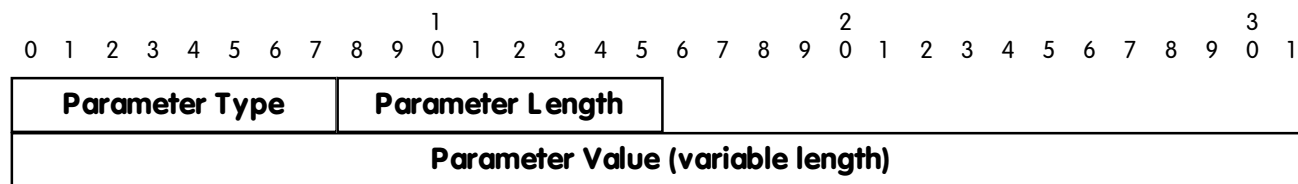
- Sent when error occurs
- BGP and underlying TCP connections closed

- KEEPALIVE message

- No message body

Contribution: Optional Parameters

- OPEN message Optional Parameter



- Type 1: Authentication Information

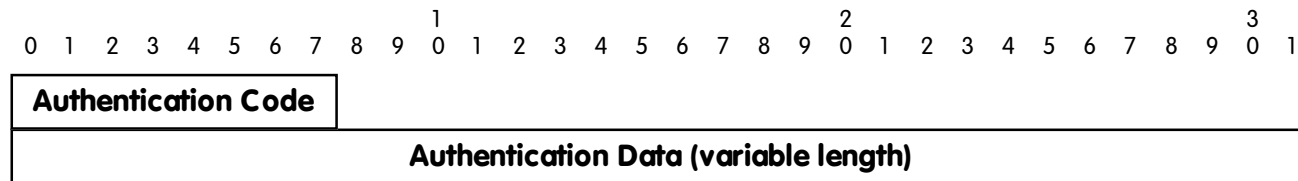
- Contains data for specific authentication mechanism

- Type 2: Capabilities Advertisement

- Communicates new features to employ during BGP peering session

Contribution: Optional Parameters (cont'd)

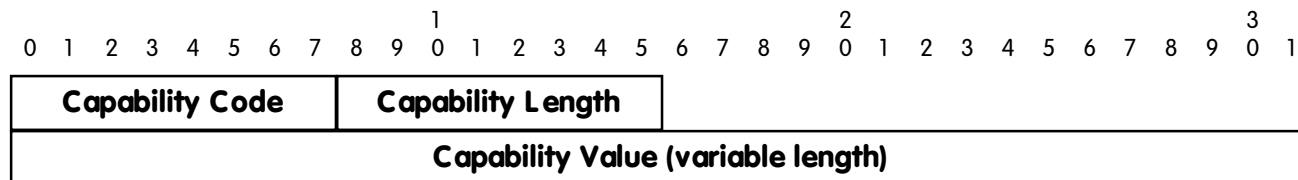
- Type 1: Authentication Information



- Used to verify identity of sender or to detect loss of synchronisation between peers
- Authentication Data contains algorithm for calculating value of Marker field
- Not yet implemented by any vendors!

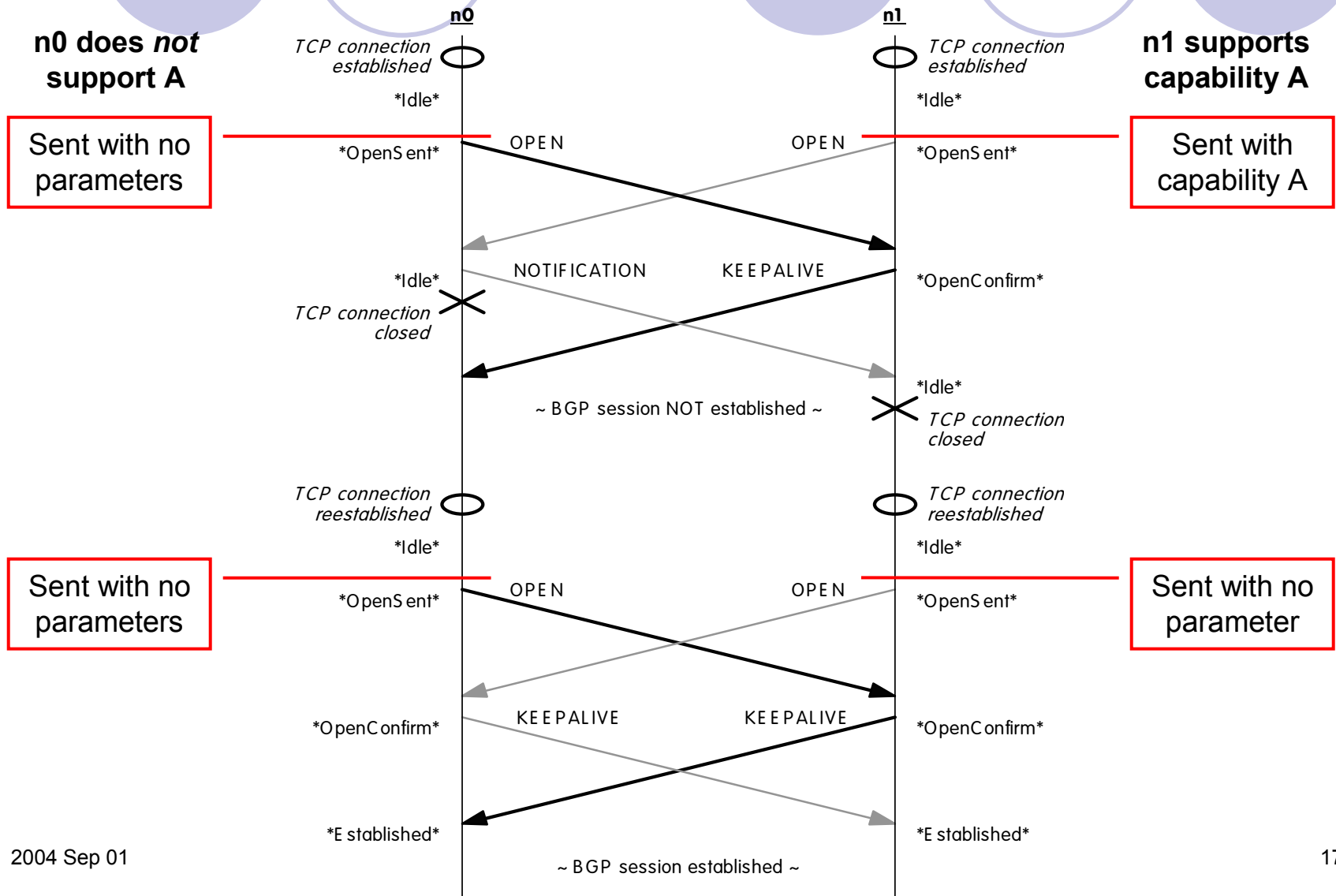
Contribution: Optional Parameters (cont'd)

- Type 2: Capabilities Advertisement



- Used to introduce new capabilities into BGP
- Allows BGP speakers to advertise supported capabilities
- Negotiated to determine which capabilities to employ during peer session

Contribution: Negotiation of Parameters



Simulation Results: Error Checking

- “Unfortunately” the simulator is ideal!
- Data is in fixed structure (object class)
 - Any message sent created directly from object
 - Any message received parsed into an object
- Errors were hard-coded to verify that the error check correctly detected the error
 - Can't show an entire scenario

Simulation Results: Error Checking (cont'd)

- Hold Time is invalid

```
Peer has received:
  Mesg type = 1, length = 29
  BGPHeader: 255-255-255-255-255-255-255-255-255-255-255-255-255-255-255-255-0-29-1
  OpenMesg: 4-0-0-0-2-10-0-0-1-0
Checking if Marker is all 1's

time: 0.0500151
peer return_ip: 10.0.1.1, peer ip_addr: 10.0.0.1
event_type: *RecvOpen*
connection_state: *OpenSent*
```

non-zero Hold Timer value is less than the minimum recommended value 3s
(current_val = 2 s)

BGP session with peer 10.0.0.1 closed.

Simulation Results: Error Checking (cont'd)

- Marker field is invalid

OPEN message

Peer has received:

Msg type = 1, length = 29

BGPHeader: 255-255-255-255-255-255-255-255-255-255-10-255-255-255-255-255-0-29-1

OpenMsg: 4-0-0-0-90-10-0-0-1-0

Checking if Marker is all 1's

The Marker field is not as expected.

BGP session with peer 10.0.0.1 closed.

Not 0xFF !

Simulation Results: Error Checking (cont'd)

- UPDATE Attribute Flags are incorrect

UPDATE message

Peer has received:

Mesg type = 2, length = 52

BGPHeader: 255-255-255-255-255-255-255-255-255-255-10-255-255-255-255-255-0-52-2

UpdateMesg: 0-0-0-25-64-1-4-0-64-2-7-2-1-0-2-192-3-7-10-0-7-1-128-9-7-10-0-7-1-24-10-0-6

Checking if Marker is all 1's

Bad flag

time: 0.251736

peer return_ip: 10.0.3.1, peer ip_addr: 10.0.6.1

event_type: *RecvUpdate*

connection_state: *Established*

optional, transitive,
partial bit values for
Path Attribute *i*

ORIGIN i=1: o=0, t=1, p=0

AS_PATH i=2: o=0, t=1, p=0

NEXT_HOP i=3: o=1, t=1, p=0 -> boooooooooo! >=(

Bad flag: NEXT_HOP

UpdateERR 4 w/ Notif: |192|3|7|10|0|6|1|

BGP session with peer 10.0.6.1 closed.

Attribute Flags error subcode

Simulation Results: Negotiation

- Syntax: *[Auth Code] / [Auth Data]*
 - Where *[Auth Data]* is string of characters
- 3 test scenarios:

n0	n1	negotiated
<none>	2/123	<none>
2/123	2/123	2/123
2/L	2/123	<none>

Simulation Results: Negotiation (cont'd)

- Scenario 1: n0 supports <none>, n1 supports 2/123

```

Simulation 0.0500116...
peer return_ip: 10.0.0.1, peer ip_addr: 10.0.1.1
event_type: *RecvOpen*
connection_state: *OpenSent*
Peer wants AuthCode = 2, AuthData = |1|2|3|
I have AuthCode = 0, AuthData = |
-|
Sending OPEN MSG 10/Length is 16.
Length of Opt Params = |6|, OPEN Mesg Length = |16|
Peer wants AuthCode = 2, AuthData = |1|2|3|
event_type: *RecvNotification*
connection_state: *Open*
Length of Opt Params = |100|, OPEN Mesg Length = |10|
OPEN MSG push is now correct
time: 0.0500151
peer return_ip: 10.0.0.1, peer ip_addr: 10.0.0.1
event_type: *RecvOpen*
connection_state: *OpenSent*
Peer wants AuthCode = 0, AuthData = |
|
Sending OPEN MSG, AuthData = |0|1|2|3|
Length of Opt Params = |0|, OPEN Mesg Length = |10|
Total length is now |29|
peer return_ip: 10.0.0.1, peer ip_addr: 10.0.1.1
event_type: *RecvOpen*
connection_state: *Open*
Length of Opt Params = |10|, OPEN Mesg Length = |10|
Peer wants AuthCode = 2, AuthData = |1|2|3|
I have AuthCode = 0, AuthData = |
|
Sending OPEN MSG with peer 10.0.1.1 closed.
Length of Opt Params = |0|, OPEN Mesg Length = |10|
Peer wants AuthCode = 2, AuthData = |1|2|3|
event_type: *RecvOpen*
  
```

Simulation Results: Negotiation (cont'd)

- Scenario 2: n0 supports 2/123, n1 supports 2/123

```

time: 0.0500235
Simulation starts
peer return ip: 10.0.0.1; peer ip_addr: 10.0.0.1
event_type: *RecvOpen*
msg_type = 4, length = 18
connection_state: *OpenConfirm*
peer return ip: 10.0.0.1; peer ip_addr: 10.0.1.1
BGP session with peer 10.0.0.1 established.
event_type: *BGPStart*

n1 connection_state: *Idle*
Peer wants AuthCode = 2, AuthData = |1|2|3|
Msg type = 4, length = 18
AuthCode = 255-255-255-255-255-255-255-255-255-255-255-255-255-255-255-255-0-19-4
Peer return ip: 10.0.0.1; peer ip_addr: 10.0.0.1
Checking if Marker is all 1's
event_type: *BGPStart*

Peer has received: *Idle*
time: 0.0500433
Msg type = 1, length = 35
peer return ip: 10.0.0.1; peer ip_addr: 10.0.1.1
n1 creating OPEN MSG w/ length 16
event_type: *RecvKeepAlive*
Inserting authentication into OPEN message:
connection_state: *OpenConfirm*
pType |1|, pLength |4|, aCode |2|, aData: 32:1 33:2 34:3
BGP session with peer 10.0.0.1 established
Length of Opt Params = |6|, OPEN Msg Length = |16|
-> Total BGP Length is now |35|
time: 29.05

peer return ip: 10.0.1.1; peer ip_addr: 10.0.0.1
n0 creating OPEN MSG w/ length 16
event_type: *KeepAliveTimerExp*
Inserting authentication into OPEN message:
connection_state: *Established*
pType |1|, pLength |4|, aCode |2|, aData: 32:1 33:2 34:3
Length of Opt Params = |6|, OPEN Msg Length = |16|
-> Total BGP Length is now |35|
time: 20.05

Peer AuthCode = 2, AuthData = |1|2|3|
Peer has received: *Established*
Msg type = 1, length = 35

Peer has received: 255-255-255-255-255-255-255-255-255-255-255-255-255-255-255-255-0-35-1
Peer wants AuthCode = 4, AuthData = 10-0-0-1-6
Msg type = 4, length = 18
AuthCode = 255-255-255-255-255-255-255-255-255-255-255-255-255-255-255-255-0-19-4
BGP Params: 11-11-11-11-11-11-11-11-11-11-11-11-11-11-11-11-0-19-4
Checking if Marker is all 1's
Using Another AuthCode to test the Marker
time: 0.0500156

time: 0.0500235
10.0.1.1; peer ip_addr: 10.0.0.1
  
```


Simulation Results: Negotiation (cont'd)

- Scenario 3: n0 supports 2/L, n1 supports 2/123

```
Simulation starts...
Peer wants AuthCode = 2, AuthData = |L|
I have AuthCode = 2, AuthData = |1|2|3|
Peer data length 10, n0 peer ip_addr: 10.0.1.1
BGP type: *BGP* peer *10.0.1.1 closed.
handle_open.auth: push to reconnect
```

```
time: 0.0500156
peer return_ip: 10.0.1.1, peer ip_addr: 10.0.0.1
event_type: *BGP*Open*
connection_state: *OpenSent*
```

```
n0:creating OPEN MESG w/ length 14
Inserting authentication into OPEN message:
I have AuthCode length 1, AuthData Code |L|, aData: 32:L
Length of Orig Params = 1, OPEN Mesg Length = |14|
Total BGP length is 10,0.0.1 closed.
handle_open.auth: push to reconnect
```

```
n0 creating OPEN MESG w/ length 16
Inserting authentication into OPEN message:
peer return_ip: length 0, peer ip_addr: aData: 32:1 33:2 34:3
Length of Orig Params = |6|, OPEN Mesg Length = |16|
Total BGP length is 16
```

```
time: 0.0500154
peer return_ip: 10.0.0.1, peer ip_addr: 10.0.0.1
event_type: *BGP*Open*
connection_state: *OpenSent*
```

```
n1:creating OPEN MESG w/ length 10
Peer wants AuthCode = 2, AuthData Length = |10|
I have AuthCode = 2, AuthData = |1|1|2|1|
```

Simulation Results: Negotiation (cont'd)

- Syntax: [*Cap Code*] → [*Cap Data*]
 - Where [*Cap Data*] is string of characters
- 2 test scenarios:

n0	n1	negotiated
<none>	1→redsox 2→mariners	<none>
1→redsox 2→mariners	2→mariners 3→angels	2→mariners

Simulation Results: Negotiation (cont'd)

- Scenario 4: n0 supports <none>
n1 supports 1→redsox, 2→mariners

```

Simulation starts...
time: 0.00027
There is nothing in the capabilities!
peer_return_ip: 10.0.0.1, peer ip_addr: 10.0.0.1
event_type: *BGPstart*
connection_state: Idle
peer_return_ip: 10.0.0.1, peer ip_addr: 10.0.0.1
connection_state: Idle*
length of Opt Params = 0
length of Opt Params = 0
OPEN Mesg Length = |10|
Total BGP Length is now |29|
peer_return_ip: 10.0.1.1, peer ip_addr: 10.0.0.1
event_type: *BGPstart*
connection_state: Idle*
length of Opt Params = 0
OPEN Mesg Length = |10|
length of Opt Params = 0
length of Opt Params = 0
n0 creating OPEN MSG w/ length 10
n1 creating OPEN MSG w/ length 32
n1 creating capability 1->redsox into NOTIF message, starting at nLen=0
inserting capability 1->redsox into OPEN message length = |8|
peer_return_ip: 10.0.1.1, peer ip_addr: 10.0.0.1
length of Opt Params = 0
OPEN Mesg Length = |10|
Data: 33:r 24:e 35:d 36:s 37:q 38:x -> pLength = |8|
event_type: *RecvOpen*
connection_state: OpenSent
inserting capability 2->mariners into OPEN message
pType |2|, cCode |2|, cLength |8|
Data: 44:a 45:i 46:l 47:n 48:e 49:r 50:s -> pLength = |10|
BGP session with peer 10.0.0.1 closed.
length of Opt Params = 0
OPEN Mesg Length = |32|
-> Total BGP Length is now |51|
time: 0.00027
n0 creating OPEN MSG w/ length 10
peer_return_ip: 10.0.0.1, peer ip_addr: 10.0.0.1
length of Opt Params = 0
OPEN Mesg Length = |10|
Total BGP Length is now |29|
connection_state: OpenSent*
BGP session with peer 10.0.0.1 closed.
time: 0.0500151
event_type: *RecvOpen*
peer_return_ip: 10.0.0.1, peer ip_addr: 10.0.0.1
event_type: *BGPstart*
peer_return_ip: 10.0.0.1, peer ip_addr: 10.0.0.1

```

Simulation Results: Negotiation (cont'd)

- Scenario 5: n0 supports 1→redsox, 2→mariners
n1 supports 2→mariners, 3→angels

```

Simulation 0.500169...
time: 0.100017, peer ip_addr: 10.0.1.1, peer ip_addr: 10.0.0.1
peer return ip: 10.0.1.1, peer ip_addr: 10.0.0.1
event type: *RecvOpen*
connection state: *Open*
peer ip_addr: 10.0.1.1
connection state: *Idle*
event type: *BGPStart*

Peer wants AuthCode = 0, AuthData = |
time: 0.100017
I have AuthCode = 0, AuthData = |
Name: | | Do not use arr auth peer ip_addr: 10.0.1.1
Event type: *BGPStart*
Peer's included in the OPEN message: 1->redsox 2->mariners
connection state: *Idle*
event type: *BGPStart*
Inserting capability 1->redsox into NOTIF message, starting at nLen=0

n0 creating OPEN MSG w/ length 22
Inserting capability 2->mariners into OPEN message
pType |2|, cCode |2|, cLength |8|
cData: 33:m 34:a 35:r 36:i 37:n 38:e 39:r 40:s -> pLength = |10|
Length of Opt Params = |12|, OPEN Msg Length = |22|
-> Total BGP Length is now |41|

n1 creating OPEN MSG w/ length 22
Inserting capability 2->mariners into OPEN message
pType |2|, cCode |2|, cLength |8|
cData: 33:m 34:a 35:r 36:i 37:n 38:e 39:r 40:s -> pLength = |10|
Length of Opt Params = |12|, OPEN Msg Length = |22|
-> Total BGP Length is now |41|

Inserting capability 3->angels into NOTIF message, starting at nLen=0
time: 0.100033
pType |2|, cCode |3|, cLength |6|
Peer return ip: 10.0.0.1, peer ip_addr: 10.0.1.1
Event type: *RecvOpen*
Peer's included in the OPEN message: 1->redsox 2->mariners 3->angels
connection state: *Open*
peer ip_addr: 10.0.1.1
connection state: *Open*
event type: *BGPStart*
Length of Opt Params = |22|, OPEN Msg Length = |32|
Total BGP Length is now |51|
Peer's included in the OPEN message: 2->mariners
peer return ip: 10.0.1.1, peer ip_addr: 10.0.0.1
    
```

Future Enhancements



- Restructuring of Optional Parameters
 - Exploit the object-oriented nature of C++
- Handling 0-bytes (NULL character) within Optional Parameter data
- Handling new Capabilities Advertisement
- Handling new Path Attributes



Conclusion

- Border Gateway Protocol (BGP) is the foremost EGP in use.
- Performance and scalability are of particular interest to the communication networks community.
- To advance the research and development of communication networks in general, simulation tools and network models have been developed, including SSFNet and ns-2.
- The purpose of this project was to expand the capabilities of ns-BGP model. Changes were BGP header and message error checking, and the implementation of and negotiation of optional parameters in the OPEN message.
- These modifications to ns-BGP will ensure that the model grows as the protocol does, with the ability to scale and handle new parameters and capabilities.



References

1. T. Bates, Y. Rekhter, R. Chandra, and D. Katz. *Multiprotocol Extensions for BGP-4*. IETF RFC 2858. June 2000. [Accessed: June, 2004.]
2. *BGP Fundamentals*. <http://www.riverstonenet.com/support/bgp/fundamentals/>. [Accessed: August, 2004.]
3. *Border Gateway Protocol*. <http://www.iana.org/assignments/bgp-parameters>. [Accessed: August, 2004.]
4. *Capability Codes – Per RFC3392*. <http://www.iana.org/assignments/capability-codes>. [Accessed: August, 2004.]
5. Cisco Systems. *Internetworking Technologies Handbook, Third Edition*. http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/bgp.pdf. Cisco Press, 2000. (ISBN: 1-58705-001-3) [Accessed: June, 2004.]
6. R. Chandra, and J. Scudder. *Capabilities Advertisement with BGP-4*. IETF RFC 2842. May 2000. [Accessed: June, 2004.]
7. R. Chandra, and J. Scudder. *Capabilities Advertisement with BGP-4*. IETF RFC 3392. November 2002. [Accessed: June, 2004.]



References

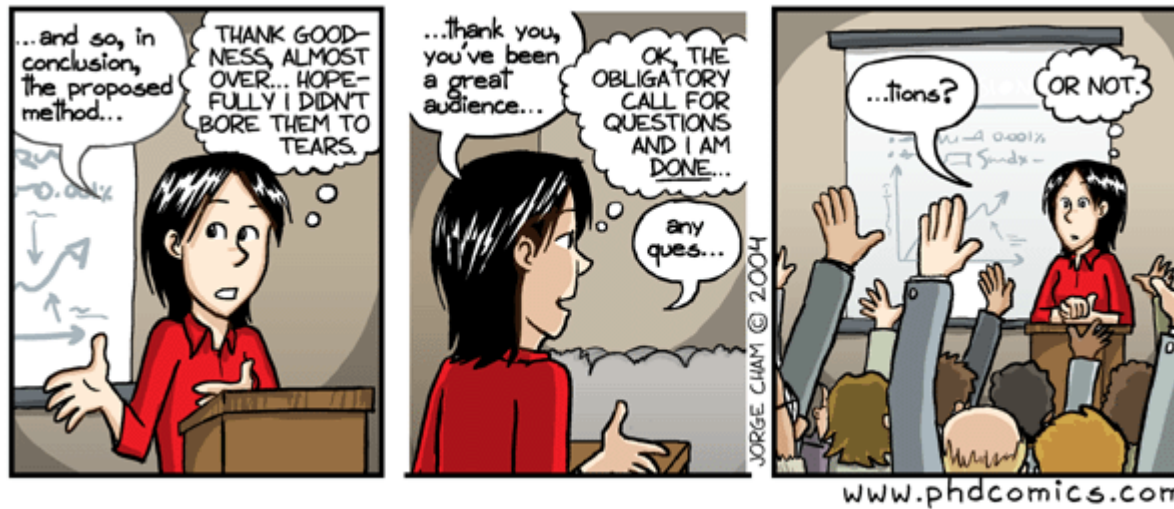
8. K. Fall, and K. Varadhan. *The ns Manual*.
http://www.isi.edu/nsnam/ns/doc/ns_doc.pdf. December, 2003. [Accessed: June, 2004.]
9. T.D. Feng, *Implementation of BGP in a Network Simulator*. M.Sc. Thesis, Simon Fraser University, April 2004.
10. T.D. Feng, R. Ballantyne, and Lj. Trajković. *Implementation of BGP in a Network Simulator*, Applied Telecommunication Symposium, ATS '04, Arlington, Virginia. [Accessed: June, 2004.]
11. M. Greis. *Tutorial for the Network Simulator "ns"*.
<http://www.isi.edu/nsnam/ns/tutorial/index.html>. [Accessed: June, 2004.]
12. *The Network Simulator – ns-2*. <http://www.isi.edu/nsnam/ns/>. [Accessed: July, 2004.]
13. I. Pepelnjak, and J. Guichard. *MPLS and VPN Architectures*. Indianapolis, IN: Cisco Press, 2001. (ISBN: 1-58705-002-1)
14. B.J. Premore. *SSF Implementation of BGP-4 v1.5.0*.
<http://www.ssfnet.org/bgp/doc/>. [Accessed: June, 2004.]

References



15. Y. Rekhter, and T. Li. *A Border Gateway Protocol 4 (BGP-4)*. IETF RFC1771. March 1995. [Accessed: June, 2004.]
16. J. Reynolds, and J. Postel. *Assigned Numbers*. IETF RFC 1700. October 1994. [Accessed: June, 2004.]
17. E.C. Rosen, and Y. Rekhter. *BGP/MPLS IP VPNs*. Internet-Draft draft-ietf-l3vpn-rfc2547bis-01.txt. September 2003. [Accessed: June, 2004.]
18. C. Semeria. *RFC 2547bis: BGP/MPLS VPN Fundamentals*. Juniper Networks white paper, part number 200012-001 03/01. Sunnyvale, CA: Juniper Networks, 2001. [Accessed: June, 2004.]
19. J.W. Stewart III. *BGP4: Inter-Domain Routing in the Internet*. Boston, MA: Addison-Wesley, 1999. (ISBN: 0-201-37951-1)
20. *Webopedia: Online Computer Dictionary for Computer and Internet Terms and Definitions*. <http://www.webopedia.com/>. [Accessed: July, 2004.]
21. P.H. Winston. *On to C++*. Reading, MA: Addison-Wesley, 1994. (ISBN: 0-201-58043-8)

Questions?



“Piled Higher and Deeper” by Jorge Cham
www.phdcomics.com