BGP Vulnerability Testing: Separating Fact from FUD

Sean Convery (<u>sean@cisco.com</u>) Matthew Franz (<u>mfranz@cisco.com</u>)

Cisco Systems Critical Infrastructure Assurance Group (CIAG) <u>http://www.cisco.com/go/ciag</u> Black Hat USA - July 2003

Agenda

Introduction

- BGP Vulnerability Testing
- Analysis of BGP Best Practices
- Active" ISP Survey
- Conclusions

If you believe what you read...

- BGP is...highly vulnerable to a variety of attacks due to the lack of a scalable means of verifying the authenticity and authorization of BGP control traffic. - S-BGP Website[1]
- Any outsider can inject believable BGP messages into the communication between BGP peers and thereby inject bogus routing information or break the peer to peer connection. draft-murphy-bgp-vuln-02.txt[2]
- Outsider sources can also disrupt communications between BGP peers by breaking their TCP connection with spoofed RST packets. - draft-murphy-bgp-protect-01.txt[3]
- The border gateway protocol...is rife with security holes and needs to be replaced, a security consultant warned. news.com[4]

Research Objectives

- Conduct a systematic analysis of BGP vulnerabilities based on testing of multiple implementations—current assumptions are largely speculative
- Measure the effectiveness of best practices in mitigating likely attacks—in the near term, hardening vendor implementations and applying best practices is all we have
- Collect data on the security posture of realworld routers and BGP implementations

Methodology

- Conduct BGP-relevant TCP attacks
- Evaluate robustness of BGP parsers using fuzztesting (similar to PROTOS)
- Conduct selected attacks in BGP Attack Tree[6] under the following conditions:
 - Blind Attacker / Non-Blind Attacker / Compromised Router
 - BGP best practices ON and OFF
- Conduct an "Active" survey of ISP best practices
 - Probe Admin ports (22/23/80)
 - Identify Permissive BGP speakers (179)

Vulnerabilities & Vulnerability Disclosure

- Three types of vulns are considered in this talk:
 - Design does what it is supposed to do
 - Implementation bug based on coding error
 - Misconfiguration weak passwords, failure to use security features, block admin ports, etc.
- Vendors have been notified of all implementation flaws
- CERT/CC has been given a set of BGP test cases to distribute to vendors
- No vendors will be identified in this talk

Attack Tree Example (Graphical)



Graphic tree representations are generated from the source attack tree.



Building on draft-convery-bgpattack-00.txt[6]

Atomic Goals

- "Compromise" MD5 Auth
- Establish unauth BGP session
- Originate unauth prefix into peer
- Change path pref of a path
- DoS BGP Session
- Spoof BGP Message

Supp. Atomic Goals

- Compromise router
- DoS router
- MITM attack
- TCP Sequence # attack
- Sniff traffic

Attack Scenarios

- Disable critical portions of Internet...
- Disable singlehomed AS
- Disable multihomed AS
- Blackhole traffic

Agenda

- Introduction
- BGP Vulnerability Testing
- Analysis of BGP Best Practices
- "Active" ISP Survey Results
- Conclusions

BGP & TCP Testing

- TCP/BGP Connection Behavior*
- TCP Resource Exhaustion*
- TCP Resets and Sequence Numbers
- MD5 (RFC 2385) Attacks
 - MD5 Dictionary Attack
 - MD5 DoS*
- Update Flooding*
- BGP Route Insertion (TCP Hijack)
- BGP Peer Hijack (ARP Spoofing)
- Malformed BGP Messages*
 - OPEN
 - UPDATE

*Conducted against multiple implementations

Testing BGP Implementations

- Goal: sample the responses of a variety of implementations to known and potential attacks
- 7 different BGP implementations were evaluated using "default" BGP configs
- When present, parenthetical notations in test result slides identify the number of implementations that exhibited that behavior
- Statistics (times/CPU utilization, etc.) were on a lightly loaded test network, so impact of certain attacks is likely to be different (greater)

Tools We Used

- Packet Generation & Injection
 - Hping[7], Nemesis-tcp[8], Netcat[9], Naptha (synsend)[10]
- Bgpcrack*
 - MD5 attacks
- TCP Test Tool (ttt)*
 - Sequence number guessing, MD5 flooding
- Tcphijack*
 - BGP route insertion

- Dsniff[11]
 - ARP spoofing
- Protocol Independent
 Fuzzer (pif)*
 - Invalid Message Generation
- Pyupdate/Pyopen*
 - Valid message generation
- "Active" ISP Survey Tools*

Some of these new tools available at:

http://www.cisco.com/security_services/ciag/tools

Connection Establishment Tests

- Identify implementation behavior during session establishment—what is necessary for successful peer negotiation? How far can the attacker get?
- How much of the message is processed and how far the state can be advanced determines risk and impact of attacks:
 - Initial SYN SYN flooding
 - Connect() ESTABLISHED/FIN_WAIT flooding
 - BGP OPEN Remote Identification/Malformed messages
 - UPDATE Route insertion/deletion

Connection Establishment (TCP)

- No standard behavior was observed across the implementations we tested
- Results varied, from least permissive (reject quietly) to most permissive (full 3-way handshake)
 - SYN from non-configured peer
 - Silent Drop (1)
 - RST-ACK (3)
 - SYN-ACK (3)
 - Spoofed SYN from configured peer (session est.)
 - RST-ACK (4)
 - SYN-ACK (3)

Connection Establishment (BGP)

- Test Results:
 - OPEN from non-configured peer
 - RST (6)
 - NOTIFICATION: OPEN Message Error/Authentication Failure (1)
 - OPEN from configured peer with invalid AS
 - NOTIFICATION: OPEN Message Error/Authentication Failure (2)
 - NOTIFICATION: OPEN Message Error Bad Peer AS (5)

Connection Establishment (BGP)

Wildcards

- Timeouts delay between session renegotiation (especially after NOTIFICATION)
 - Delay of 1-3 minutes before new connection (4)
 - No timeouts (3)
- Send OPEN immediately after reaching established state (1)
- No implementation allowed BGP OPENs with the wrong AS or from non-configured peer to reach BGP ESTABLISHED state—as a result, *TCP spoofing is required to inject data*

TCP Resource Exhaustion vs. BGP

- Goal: prevent new BGP sessions from being established or impact existing sessions
- Why: many BGP implementations are tightly integrated with TCP stacks and there may be "collateral damage"
- Should be the easiest to conduct and require the least amount of knowledge and access
 - SYN Flooding
 - ESTABLISHED Flooding
 - FIN_WAIT1 Flooding

SYN Flooding

Exhaust number of sessions in SYN_RCVD state

Attacker# synsend 10.89.168.101 10.89.168.89 1 Randomizing port numbers. Sending SYN packets.

Victim# netstat -an | grep --tcp tcp 0 0 10.89.168.101:179 10.89.168.99:4189 SYN_RECV tcp 0 0 10.89.168.101:179 10.89.168.99:8017 SYN_RECV tcp 0 0 10.89.168.101:179 10.89.168.99:56477 SYN_RECV tcp 0 0 10.89.168.101:179 10.89.168.99:41185 SYN_RECV

ESTABLISHED Flooding

Stress peer establishment or overflow socket file descriptors

```
Attacker# synsend 10.89.168.101 10.89.168.89 1
Randomizing port numbers.
Sending SYN packets.
Attacker# srvr -SAa 10.89.168.10
Victim# netstat -an | grep --tcp
tcp 0 0 10.89.168.101:179 10.89.168.99:36601 ESTABLISHED
tcp 0 0 10.89.168.101:179 10.89.168.99:59545 ESTABLISHED
tcp 0 0 10.89.168.101:179 10.89.168.99:49340 ESTABLISHED
```

FIN_WAIT 1 Flooding

Stress peer deletion or exhaustion of socket file descriptors

```
Attacker# synsend 10.89.168.101 10.89.168.89 1
Randomizing port numbers.
Sending SYN packets.
Attacker# srvr -SAfa 10.89.168.10
Victim# netstat -an | grep --tcp
tcp 0 1 10.89.168.101:179 10.89.168.99:35734 FIN_WAIT1
tcp 0 1 10.89.168.101:179 10.89.168.99:15142 FIN_WAIT1
tcp 0 1 10.89.168.101:179 10.89.168.99:56006 LAST_ACK
tcp 0 1 10.89.168.101:179 10.89.168.99:63718 LAST_ACK
```

TCP Resource Exhaustion vs. BGP Results

- Goal was to just impact TCP and as a result, BGP—we know there are infinite ways to kill a box (saturate links, punt to CPU, fill non-TCP queues, etc.)
- Impact to implementations that SYN/ACK peers (or when spoofed)
 - Up to 5-6 minute delay in BGP session establishment – peers under attack could negotiate outbound sessions with other peers
 - Moderately elevated CPU utilization and latency
 - No impact on existing sessions

TCP Resource Exhaustion Results

The bottom line

- An attacker would have to find a way to break the current session and SYN flood both peers (and possibly spoof the src, depending on the implementation) to cause significant impact
- Implementations that allow state past SYN_RECVD may have issues—but ACLs can mitigate this—blind connect() spoofing is hard

TCP Resets and Sequence Number Guessing

- Successful TCP resets require a valid 4-tuple and sequence number (not ttl)
- TCP Test Tool (ttt) is able to generate messages easily assuming local access to the wire:

```
18:22:59.328544 99.0.0.3.179 > 99.0.0.5.32324: P
272350230:272350249(19) ack 4142958006 win 15531: BGP
(KEEPALIVE) [tos 0xc0] [ttl 1]
18:22:59.527079 99.0.0.5.32324 > 99.0.0.3.179: . ack
272350249 win 15543 [tos 0xc0] [ttl 1]
```

./ttt -T 2 -D 99.0.0.5 -S 99.0.0.3 -x 179 -y 32324 -fR -s 272350249

May 1 18:23:13.425: %BGP-5-ADJCHANGE: neighbor 99.0.0.3 Down Peer closed the session

Nothing new here. Tcpkill (from dsniff) works, too.

TCP Resets Results

The peer is fully reestablished in 50 seconds (test network) - several minutes (production network):
 May 1 18:24:50: %BGP-5-ADJCHANGE: neighbor

99.0.0.5 Up

- Various research [12], and [13] have found flaws in some implementations of TCP ISN selection. This should be a solved problem for most implementations though (did not repeat tests).
 - This research depends upon access to a range of initial sequence numbers from the router (we can prevent this with BCPs).
- If implementations went with pseudo-random source ports the number space moves from 2³² to 2⁴⁸.

TCP Resets Time Requirements

A theoretical blind attack @ 1 million pps ~ 30 minutes to just guess the seq. number (assuming a correct guess after iterating through 50% of the space).

 $(2^{32}/2)/1,000,000 = #$ of seconds

- Our tool was able to generate 62,500pps* ~ 9 hours
- Since the attacker won't know which side is 179 vs. a high port multiply these numbers by 2
- With source port randomization, this goes to 4 years in the first example (1 mil. pps to guess 1 48 bit number and 142 years assuming 62,500pps and needing to guess both sides):

 $((2^{48}/2)/62,500)x^2 = #$ of seconds

*What sort of event is 62.5kpps on *your* router?

TCP Reset Conclusions

- Blind TCP seq. guessing is operationally impossible with a router using BCPs – with proper RFC 2827[14] filtering, the packet won't even reach the destination
- Even without BCPs, this is quite a lot of work for 50 seconds (up to 5 minutes?) of down-time
- A successful TCP reset attack would need to be constantly repeated to keep a session down and would need to be duplicated on many routers to cause substantial impact to the Internet's routing tables
- Any TCP sequence number attack will require lots of packets potentially causing link saturation or other problems (routers should notice)

MD5 Dictionary Attack

- All the information needed to compute RFC2385[15] MD5 authentication is present in the packet except the secret itself:
 - TCP Pseudo-header (sIP, dIP, protocol number, segment length)
 - TCP header (w/o options, and 0 checksum)
 - TCP Segment data (if any)
- "Bgpcrack" test tool uses .pcap files and a dictionary file (with permutation definitions) or can increment through all possible passwords using John the Ripper[16]
- Tool can also run in "online" mode by sending a segment repeatedly with different MD5 passwords—allowing remote brute force (similar to Telnet/HTTP attacks)



```
elapsed time = 8 seconds
```

- A permuted version of the above password "DOM1NO" was found in 3.5 hours with no dictionary file as help: "./john -stdout:6 -incremental | ~/bgpcrack-2.0/bgpcrack -r ~/md5cap3 -w - n 1 port bgp -R ~/bgpcrack-2.0/rules.ini"
- Countermeasures: Choose strong passwords: draftietf-idr-md5-keys-00.txt[17]

MD5 Testing

- Test Combinations
 - Valid or invalid peer
 - Established or non-established session
 - Valid or invalid password
 - TCP SYN, PSH-ACK, RST
- Two possible results: drop silently or RST
- Implementations that dropped silently had lower CPU impact than those that RST
- Worst attack using MD5—SYN-Flooding from peer if no session established (70%)
 - Dropped to 30-40% if session already established

MD5 Flooding Results

- Order of processing impacts results
 - Some processed MD5 before sequence number resulting in greater CPU impact when flooded
 - Others processed TCP (checked for valid ports, sequence numbers) resulting in lesser impact
 - TCP behavior (especially with regard to existing session) impacts results

BGP Update Flooding

 Wrote python script to establish session and continue to add an arbitrary number of routes at will

bash-2.05a\$ pyupdate 192.168.1.200 100 eth0

```
Source IP: 192.168.1.101
Connecting to 192.168.1.200 (45 bytes received)
Sending keepalive...
How many routes to send? 10000
Split into 1000 route updates?y
Generating 10000 routes (40000 bytes)
Building UPDATE...
Source IP: 192.168.1.101
Routes: 1000
NLRI: 4000
BGP Length: 4048
```

BGP Update Flooding Results

- Variations among implementations:
 - Rate at which new routes could be processed
 - CPU Utilization and ICMP latency
 - Behavior when route ceiling was hit
 - Will not accept new routes
 - Tears down BGP session
 - Overwrites old routes

BGP Route Insertion (TCP Hijack)

- Assuming the ability to guess the TCP sequence number; routes can be inserted using a single spoofed update message.
- As soon as the real BGP speaker communicates again (keepalive), an ACK storm ensues due to the overlapping sequence numbers.
- In our testing we found that the ACK storm takes about 5 minutes to resolve during which time the spoofed route will remain in the table and be passed to other routers.

BGP Route Insertion (cont.)

- TCP hijack will insert a binary payload by listening to the sequence numbers on the wire.
- If the attacker stays inline (via ARP or MAC spoofing) the route could stay longer. There may be ways to back-out gracefully without killing the existing session (further research warranted).

5wld: BGP(0): 99.0.0.5 rcvd UPDATE w/ attr: nexthop 99.0.0.5, origin i, metric 0, path 5 5wld: BGP(0): 99.0.0.5 rcvd 7.7.7.0/24

BGP Peer Hijack (ARP spoof)

- Using arpspoof an attacker can easily poison the ARP table of a BGP peer and cause the session to be terminated and reestablished with the attacker.
- By spoofing only one peer of the victim both the real BGP speaker and the victim will remain connected. (the victim still peers with other ISPs)


Protocol Fuzzing using PIF

- Provide a general purpose engine to generate malformed fields deeper into packet than existing tools such as ISIC
- Allow a large number of messages for many protocols to be quickly and easily generated without completely describing the protocol
- Focus on complex Type-Length-Value protocols such as BGP and IKE where implementation errors are likely

PIF: Basic Principle of Operation

- The deeper into the message we are able to inject invalid data, the greater confidence we have that the implementation will properly parse malformed input
- This will find improper handling of incorrect length values, truncated messages, and illegal type codes which can cause unstable operation

Message/Packet Depth



PIF Components

Protocol Description Language (PDL)

- Describes possible message syntax
- Consists of a flat-file tree that is chained together
- Each file is a "block" discrete protocol unit that consists of multiple fields (line within file)

User Input Module

 Parses protocol descriptions and instantiates subset of protocol messages to be generated

Result is protocol "template" which is passed to generator

- Message Generation Module
 - Creates final binary output based on template
- Injection Scripts
 - Inject at TCP, UDP, IP, Ethernet layer

Sample Fuzzer run for BGP

ciag-530b:~/pif/pdl/bgp# pif bgp build fuzz

```
====>bqp.pdl<====
marker> fixed field, no input required
[value] [s]hort [l]ong [z]ero [r]andom [v]alid or e[x]it
bqp len>v
        Using a valid length, calculating at fuzz time.
['0x04', 'keepalive', '0x01', 'open', '0x02', 'update', '0x03',
   'notification']
[c]ycle [value] [p]ermute [r]andom [s]weep [z]ero e[x]it
bgp type>open
====>bgp-open.pdl<====
ver> fixed field, no input required
                [value] [p]ermute [r]andom [s]weep [z]ero e[x]it
my as>100
```

From protocol description to identified flaw



Malformed OPEN Testing

- Generated 100 test cases for each "layer" using pif "backtrace" function
- Messages were from completely invalid to mostly valid:
 - Completely Random
 - Valid Marker + fuzzload
 - Valid Length + fuzzload
 - Valid Version (4) + fuzzload
 - Valid AS + fuzzload
 - Hold Time + fuzzload
 - Identification + fuzzload
 - Random Option Parameters

Sample Malformed OPEN

	⊞ Trai	nsmi	issi	ion	Cor	ntro	ol F	rot	ocol,				-			-		Dst	Port	: bg	p (17	(6, 15
		Inkn			_																	
						byt																
			_			13 k	-															
)	/pe:	: Ur	hkino	own	Mes	sag	je (18	34 J)											
í	<u> </u>																					
ľ	<u> </u>																					
	0000	00	00	00	00	00	00	00		00	00	00	00	08	00		00				E.	
	0010	01	40	5a	70	40	00	40		21	45	7f	00	00	01		00	.@Zp@	a.e.	.EO.	0.	
	0020	00	01	80	8f	00	b3	d3		3b	e6	d3	99	de	6e		18		M	;	.n	
	0030	7f	ff	99	d0	00	00	01		98	0a	00	3d	61	8b		3d	0. <u></u>		=	a=	
	0040	61	8b	ΤŤ	<u>+</u> +	++	††	++		Ħ	++	++	++	††	++	ff	<u>††</u>	a	• • •	• • • •		
	0050	ff	ff	of	03	b8		10		<u>14</u>	b6	d9	e1	40			f2		•••	····'		
	0060	0f	a5	bd	f8	97	7e			30		49	e9							0.1.		
	0070	4b	fd	38	81	64 41					a2 ch				bb		d2	K.8.(r		
	0080	6a 14	e4 4 d	00 a4	72	20		af		:14 ≘a	6b 30	47 7e	bd a8	C4 68	ba dd	6f 88	28 44			.kG.		
	0090 00a0	07	4u C3	ач С5	ei 6e		fc			⊒a 2d	30 a8	ad	ао 4е	00 8e			44 b4	.M,		.0~. N		
	loobo	fź	23	Ze			11			28 28			4e f8	79			91	• • • • ! ! · • #~• • ·	•••	N		
	00000	b5	44	07	8ř			94					1d	74	31		e4	.D		··· <u>y</u> ··		
	lõõdõ	96	29	33	d0	46	89	95		11	9d	41	81	69	74	_	ed	.)3.F		<u>A</u> .	_	
	00e0	72			13	ea	Ť1			δĖ	5e	68	ea	ŠÕ			17	-	.p.			
	loofo	ca	ēd	37	93		31			2e	64	c2	60	ć2	cĒ		a9		1>.	.d.`		
	0100	e9	36	97	fd		1b					bd	26	38	68		ad		Т	&		
	0110	59	fd	61	0d		e2			25	bf	9b	66	76			f9	Y.a		f		
	0120	98	08	65	db	46	d1	60	f2 (25	ea	e6	82	f6	58	13	72	e.P	÷.`.			
	0130	01	с9		fb		87			5d	97		eb	7c		0d	26			m.5.	e.&	
	0140	4 d	37	5e	fa	cb	6c	c8	01 (03	09	b2	61	62	13			М7∧	٦	a	b.	

Another Malformed OPEN

			.,										<i>,</i> ,			
🖽 Tra	nsmissio	n Con	trol I	proto	ocol,	src	Por	t:	330	67	(33	067),	Dst	Port:	179	(179)
🖾 Bor	der Gatev	way P	rotoc	51												
	DPEN Mess	age														
	Marker	: 16	bytes													
	Length	: 169	byte:	5												
	Type: OPEN Message (1)															
	Version				-											
	My AS:	4128	9													
	Hold t															
	BGP ide			188.8	37.220	.251	L									
	Optiona							rtes								
	⊟Optiona						,									
					arame	ter										
	Unknown optional parameter Unknown optional parameter															
, [2]																
_																
0000	00 04 8				03 47			08	08	00		00	-	J G		Ε.
0010	00 d1 6	d e8	40 00	40 (06 48	3 27	C0	a8	Ō1	63	C0	00 a8	m.(a.a. ⊦	· «	
0010	00 d1 6 01 64 8	d e8 1 2b	40 00 00 b3	40 (5a 3	06 48 35 1f	3 27 98	C0		Ō1		C0	00	m.(.d.+	8.0. ⊦ <u>z5</u> .		
0010 0020 0030	00 d1 6	d e8 1 2b	40 00 00 b3 00 00	40 (5a 3 11 1	06 48 35 1f	27 98	c0 22	a8 9e	01 75	63 a7	c0 50	00 a8 18	m.(.d.+ <u>g</u> .	8.@. ⊦ z5 .	¦'∢ .".u.	P.
0010 0020 0030 0040	00 d1 6 01 64 8 16 d0 6	d e8 1 2b 7 b5	40 00 00 b3 00 00	40 (5a 3 111 1 00 a	06 48 35 1f 17 11 49 01	27 98 11 55	c0 22 ff a1	a8 9e ff 49	01 75 11 3a	63 a7 ff 70	c0 50 11 bc	00 a8 18 ff 57	m.(.d.+ g.	9.@. ⊦ 	¦'∢ .".u.	P.
0010 0020 0030 0040 0050	00 d1 6 01 64 8 16 d0 6 ff ff f dc fb 2	d e8 1 2b 7 b5 f ff 6 37	40 00 00 b3 00 00 ff ff 07 f7	40 (5a 3 ff f 00 a 47 8	06 48 35 1f 1f ff a9 01 36 06	27 98 ff c5 49	c0 22 ff a1 1f	a8 9e ff 49 f0	01 75 71 3a b3	63 a7 ff 70 9f	c0 50 ff bc a8	00 a8 18 ff 57 0f	m.(.d.+ g.	8.@. ⊢ Z5	¦'∢ .".u.	P.
0010 0020 0030 0040	00 d1 6 01 64 8 16 d0 6 ff ff f dc fb 2 20 89 2 43 5b a	d e8 1 2b 7 b5 f ff 6 37 a a3	40 00 00 b3 00 00	40 (5a 3 ff f 00 a 47 8 53 7	06 48 35 1f ff ff a9 01 36 0e 7e 7e	27 98 ff c5 49 d0	<pre>c0 22 ff a1 1f e6</pre>	a8 9e ff 49 f0 c9	01 75 75 3a b3 e5	63 a7 ff 70 9f ae	c0 50 ff bc a8 ce	00 a8 18 ff 57	m.(.d.+ g.	8.@. ⊢ 	¦'∢ .".u.	P.
0010 0020 0030 0040 0050 0060 0070 0080	00 d1 6 01 64 8 16 d0 6 ff ff f dc fb 2 20 89 2 43 5b a 73 4a d	d e8 1 2b 7 b5 f ff 6 37 a a3 6 e1 d 81	40 00 00 b3 00 00 ff ff 07 f7 c9 8f 75 f3 2f ec	40 (5a 3 111 1 00 a 47 8 53 7 5d 2 2a (06 48 35 1f ff ff 39 01 36 0e 7e 7e 2c 04 0f 48	27 98 11 298 15 298 49 200 49 49 40 40 40 40 40 40 40 40 40 40 40 40 40	C0 22 ff a1 1f e6 4e 63	a8 9e ff 49 f0 c9 7a d6	01 75 75 3a b3 e5 cc d5	63 a7 ff 70 9f ae 8d c8	C0 50 ff bc a8 ce 79 31	00 a8 18 ff 57 0f d7 27 73	m.(.d.+ g.	8.@. ⊢ 	('(.".u. I:;	P.
0010 0020 0030 0040 0050 0060 0070 0080 0090	00 d1 6 01 64 8 16 d0 6 ff ff f dc fb 2 20 89 2 43 5b a 73 4a d 8d 4a c	d e8 1 2b 7 b5 f ff 6 37 a a3 6 e1 d 81 9 50	40 00 00 b3 00 00 ff ff 07 f7 c9 8f 75 f3 2f ec e6 50	40 (5a 3 11 1 00 a 47 8 53 2 54 2 2a 0 d8 9	06 48 35 1f 17 ff 19 01 36 06 7e 76 2c 04 0f 48 97 3a	27 98 61 25 249 249 249 264 364	c0 22 ff 1f e6 4e 63 ec	a8 9e ff 49 f0 7a d6 01	01 75 3a b3 e5 c0 d5 a0	63 a7 70 9f ae 8d 7d	c0 50 ff bc a8 ce 79 31 ea	00 a8 18 ff 57 0f d7 27 73 22	m.(.d.+ g. &7 &7 	3.@. ⊢ Z5 G S~ ~ u.], . /.*. ⊢	¦'(.".u. 	P.
0010 0020 0030 0040 0050 0060 0070 0080 0090 0090 00a0	00 d1 6 01 64 8 16 d0 6 ff ff f dc fb 20 20 89 2 43 5b a 73 4a d 8d 4a c 89 6d e	d e8 2b 7 b5 f f7 a a3 6 e1 9 50 f 6f	40 00 00 b3 00 00 ff ff 07 f7 c9 8f 75 f3 2f ec e6 50 69 a9	40 (5a f 00 a 47 a 53 a 2 a 8	06 48 35 1f 7f ff 39 01 36 0e 7e 7e 7e 7e 2c 04 0f 48 97 3a 3e ba	27 98 77 298 49 49 40 40 40 40 40 40 40 40 40 40 40 40 40	C0 22 ff a1f e6 ec ec	a8 9e ff 49 f0 7a d6 01 0e	01 75 75 83 85 85 85 85 85 85 85 85 85 85 85 85 85	63 a7 ff 9f a8d c8 c8 c8 c8 c8 c8 c8 c8 c8 c8 c8 c8 c8	C0 50 ff bc ace 79 31 ea 78	00 a8 18 ff 57 of d7 27 73 22 75	m.(d.+) g. &7 	8.@. ⊢ 	l'(.".u. I.I:; .NZ	P.
0010 0020 0040 0050 0060 0070 0080 0090 0090 0040	00 d1 6 01 64 8 16 d0 6 ff ff f dc fb 20 20 89 2 43 5b a 73 4a d 8d 4a c 89 6d e 83 57 b	d e8 2b 7 b5 6 37 6 a3 6 e1 9 50 f 03 f 03	40 00 00 b3 00 00 ff f7 c9 ff c9 ff c9 f3 c9 f3 c9 a9 c9 0b	40 (5a 3 ff f 00 a 47 8 53 7 2a (d8 9 83 0 ef 8	06 48 35 1f 7f ff 39 01 36 06 7e 7e 7e 7e 2c 04 0f 48 37 3a 37 06	27 98 ff c5 d0 f4 e9 64 e9 ede	c0 22 ff a1f e6 4e 63 ec 5b	a8 9e ff 49 f0 7a 00 bd	01 75 75 83 85 85 85 85 85 85 85 85 85 85 85 85 85	63 a7 f70 f ed 80 80 93	C0 50 ff bc a8 ce 79 31 ea 78 56	00 a8 18 ff 0f d7 27 73 22 7b 1f	m.0 d.+ g. &7 .* 	8.@. ⊢ 	1' 	P.
0010 0020 0030 0040 0050 0060 0070 0080 0090 0080 0090 0080 0090 0080	00 d1 6 01 64 8 16 d0 6 ff ff f dc fb 2 20 89 2 43 5b a 73 4a d 8d 4a c 89 6d e 83 57 b d7 a3 e	d e8 2b 7 ff 37 6 37 6 37 6 37 6 37 6 37 9 6 7 9 7 9 7 9 7 9	40 00 00 b3 00 00 ff f7 c9 ff e0 c9 ff e0 69 0b 56 0b	40 (5a 2 ff f 00 a 47 8 53 2 2a (d8 9 83 0 ef 8 8e 1	06 48 35 1f 7f ff 39 01 36 06 7e 76 2c 04 0f 48 37 06 13 27	27 98 ff c5 d0 f4 e9 64 e9 de d9 de d9	c0 22 ff a1f e6 46 63 ec 55 52	a8 9 1 4 9 7 4 9 7 4 0 9 4 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	01 75 17 a 3 5 c 5 0 a d f 5 6	63 a7 f70 f ed 8 c70 63 61	C0 50 ff a8 c9 31 e78 56 d1	00 a8 18 ff 57 of d7 27 73 22 75	m.0 d.+ g. &7 .* 	8.@. ⊢ 	1' 	P.
0010 0020 0040 0050 0060 0070 0080 0090 0090 0040	00 d1 6 01 64 8 16 d0 6 ff ff f dc fb 20 20 89 2 43 5b a 73 4a d 8d 4a c 89 6d e 83 57 b	d e8 2b 7 ff 37 6 37 6 37 6 37 6 37 6 37 9 6 7 9 7 9 7 9 7 9	40 00 00 b3 00 00 ff f7 c9 ff c9 ff c9 f3 c9 f3 c9 a9 c9 0b	40 (5a 2 ff f 00 a 47 8 53 2 2a (d8 9 83 0 ef 8 8e 1	06 48 35 1f 7f ff 39 01 36 06 7e 76 2c 04 0f 48 37 06 13 27	27 98 ff c5 d0 f4 e9 64 e9 ede	c0 22 ff a1f e6 46 63 ec 55 52	a8 9 1 4 9 7 4 9 7 4 0 9 4 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	01 75 173 25 25 25 25 25 25 25 25 25 25 25 25 25	63 a7 f70 f ed 80 80 93	C0 50 ff a8 c9 31 e78 56 d1	00 a8 18 ff 0f d7 27 73 22 7b 1f	m.0 d.+ g. &7 .* 	8.@. ⊢ 	1' 	P.

Malformed BGP Update Testing

- Generated 100 test cases for each set:
 - Valid BGP type (UPDATE) + fuzzload
 - Valid BGP type (UPDATE with invalid BGP length)
 + fuzzload
 - Unfeasible length (set to 0) + fuzzload
 - Valid Path Attribute Length + fuzzload
- These test cases provide less comprehensive coverage than OPENs and more testing may be necessary

Sample Malformed BGP Update

	nsmission Control Protocol, Src Port: 33730 (33730), Dst Port: 17	79 (179									
	ler Gateway Protocol										
<u> </u>	PDATE Message										
	Marker: 16 bytes										
	Length: 2495 bytes										
	Type: UPDATE Message (2) Upfoacible neutre length: 10606 bites										
	Unfeasible routes length: 19606 bytes ∃Withdrawn routes:										
L (Withdrawn route length 214 invalid										
ΓUni	eassembled Packet: BGP]										
0040	76 17 if if i										
0050	ff ff 09 bf 02 4c 96 d6 d9 ce cc 96 5c a2 df f6L	$\setminus \dots$									
0060	40 e8 40 a9 d5 41 3b bd f1 32 7b e3 ce 27 cb d7 0.0A;2{.										
0070	93 f7 7d 01 f8 51 d5 cb a8 bf 37 8f 5f 53 44 b7}o7 dc 99 31 8c 42 55 b0 35 88 ac 64 22 e2 31 7f 5f1.BU.5d	SD. ".10									
0090		.h=9`									
00a0	ab db c3 05 23 38 53 32 f5 b0 6d 89 31 f9 49 9f#8s2m	.1.I.									
00b0	65 fd bb 31 f1 47 cd 77 44 e4 f4 23 ac d2 0a d8 e1.G.w D	₩									
00c0 00d0	f1 54 60 86 6e d9 dc 7b bd da c5 ab 85 0b 2e f9 .т`.n{ 4b dd 40 b9 25 e6 bb b3 63 a7 b9 13 1b b8 a1 df к.@.% с										
00e0	$36 \ 66 \ de \ 3f \ 22 \ 71 \ e6 \ 7e \ 45 \ aa \ 26 \ 8c \ 9f \ 9a \ 62 \ b6 \ 6f.?"q.~ E.&$	b.									
00f0	f5 94 f9 88 01 46 8e aa 80 85 a5 e4 82 c3 6b d9 F	k.									
0100	9f 0d 4c c7 de bc c5 bb 9c d6 d1 3b 33 d4 3f 03 L; d1 5a 23 d8 34 cc 29 7b d3 d5 f2 93 78 4c aa f9 z#.4.){	;3.?.									
0120	d1 5a 23 d8 34 cc 29 7b d3 d5 f2 93 78 4c aa f9 .z#.4.){ aa 5c 23 5a 62 a6 17 76 b8 56 93 ee 2c 87 a0 a7 .\#zbv .v										
0130	ad 10 88 9a a0 89 c3 95 05 8c d2 69 8d 76 d2 a9	i.v.									
0140	ce b3 c3 1f b5 a1 f4 f9 f5 79 f3 f2 5d ef f4 07y.	.]									
0150	6f f1 95 ee 50 89 db 22 44 cf 27 e7 1a 74 61 ab oP" D.'. 4b 05 8b f0 70 0a 5e dc 37 50 33 8b 27 87 22 c7 Kp.^. 7P3.	ta.									
0170		6JU.									
0180	b1 3d fe a5 43 2a d4 27 Oc c8 8d e2 a6 3d 58 cb .=c*.'	=X.									
0190		.j6.r									
01a0	41 7e 75 b5 5e e7 71 bf fc 17 59 0e 71 57 a8 72 A~u.A.dY.	.aw.r									

BGP Malformed Message Results

- Based on 1200 test cases:
 - Only 4 different flaws were found impacting 4 of the 7 implementations tested (flaws were unique to each implementation)
 - 3 of the flaws required the attacker to be a valid configured peer and/or valid AS

Areas For Further Testing

- Need more comprehensive set of test cases for UPDATE
 - iBGP testing vs. eBGP testing
 - Malformed update propagation issues
 - Reproduce our tests to confirm results

BGP/TCP Implementation Recommendations

- Extensive, configurable logging of connection failures (TCP, BGP, MD5)
- Aggressive rejection of TCP connections from non-configured peers and aggressive timeouts can minimize TCP resource exhaustion attacks
- Aggressive rejection of unauthorized (invalid peer and AS) can minimize the impact of most remote non-blind attacks
- Consider source port randomization
- Lengthy BGP session timeouts (i.e. 60 seconds) can minimize message flooding attacks
- Implement the BGP TTL Hack[18]

Agenda

- Introduction
- BGP Vulnerability Testing
- Analysis of BGP Best Practices
- "Active" ISP Survey Results
- Conclusions



Test summary w/No BGP BCPs

Blind Attacker

- Due to TCP sequence guessing requirement, most attacks are practically impossible
- Everything depends on getting access to a link with BGP speakers or compromising a router

Non-Blind Attacker

- Sessions reset at will
- Routes inserted (but ACK storm resets the session shortly)
- Peer hijacking is possible with ARP spoofing
- Compromised Router
 - Tear down sessions, insert invalid routes, modify attributes (could require a rogue implementation), reconfigure to allow malicious peering.

BGP BCPs For Tests

- Based on basic router best practices and Rob Thomas' BGP Hardening Template[19] and ISP Essentials[23] (additions in red)
 - Unicast RPF (RFC 2827 Filtering)
 - Ingress and Egress Prefix Filters (with max prefix length limit and bogon filtering)
 - Route Flap Dampening
 - Bogon route filtering
 - BGP Network ACLs
 - TCP MD5 (with strong passwords)
 - Static ARP for Ethernet peering
 - Static CAM entries and port security [20] for IXP Ethernet switches
 - AS Path Filtering not tested (needs more research)

Key BGP BCPs

Blind Attacker

- RFC2827 even without broad adoption, you can prevent people from spoofing your ranges, and thus all TCP attacks
- BGP ACLs Don't let invalid BGP packets on the wire
- Non-Blind Attacker
 - L2 best practices stops sniffing, hijacking, etc.
 - MD5 adds additional pain to the attacker
 - Ingress / Egress prefix filtering limits damage in case of compromise (update flooding, etc.)
- Compromised Router
 - Ingress / Egress prefix filtering limits extent of damage a compromised router can cause (update flooding, etc.)

BGP BCP Analysis Summary

- As expected, a compromised router is the most beneficial asset to an attacker in a network with BGP BCPs
- TCP MD5 is helpful everywhere, but is particularly useful in shared media environments (deployment issues are being worked on)
- L2 Best practices matter in shared media environments
- Packet filtering to stop spoofed BGP messages at your edge and on each peer will prevent almost all TCP based attacks—and as a result almost all BGP based attacks from non-compromised routers

Agenda

- Introduction
- BGP Vulnerability Testing
- BGP BCP Analysis
- Active" ISP Survey Results
- Conclusions

Test Methodology

- Goal was to non-intrusively assess basic BCP adoption through probes from an arbitrary IP address
 - Limit scanning to prevent production impact—a single SYN with no retries
- Build table of potential BGP speakers by running traceroutes to approx. 120,000 hosts (one for each CIDR block in the Internet's route table)
- Probes:
 - Send 1 x TCP SYNs to ports 22, 23, 80, 179
 - Embed message in payload identifying probes as nonmalicious
 - Measure response (SYN ACK, RST, No Response)
- Send BGP OPEN to those that SYN-ACK on port 179
 - Sessions used an unused AS #
 - Record BGP message that is returned

"Active" ISP Survey Results (Summary)

- Total non-1918 routers probed: 115,466
- BGP Speakers
 - SYN-ACK 4,602
 - RST 3,088
 - No Response 107,777
- BGP Open Test Results
 - OPEN / NOTIFICATION 1,666
 - AUTH FAIL 1635
 - CEASE 11
 - BAD AS 20
 - NOTIFICATION ONLY 84
 - AUTH FAIL 1
 - CEASE 83
 - RST 264
 - Connect (No Data) 2,147

SSH daemons: 6,349
Telnet daemons: 10,907
HTTP Servers: 5,565
16,815 routers were reachable* on at least one admin interface (14.5% of probed routers)

*Based only on receipt of SYN-ACK, so daemons that you can actually connect() to could be lower!

Admin Port Reachability (by Country)

Several countries had either 100% of their routers accessible or 0% but were not counted since there were less than 10 routers probed in each of these countries.

Honorable Mentions:

Spain - 878 (5.13%) **France** - 1820 (6.48%) **Great Britain** - 4005 (7.72%)

Country	Total Probed Routers	Percentage Admin Reachable
Maldives	10	0%
Gibraltar	16	0%
Iceland	34	2.94%
Kazakstan	80	3.75%
Fiji	23	4.35%
USA	56481	14.22%
Average		14.5%
Canada	4555	15.32%
Kyrgyzstan	19	52.63%
French Polynesia	12	58.33%
Tanzania	10	60%
Uzbekistan	25	68%
Bahamas	15	73%

Conclusions

- The most damaging attacks are caused by the deliberate misconfiguration of a trusted router
 - Compromising the router is not BGP specific and is not covered here. Best practices should be well understood for router hardening[5]
- Assumptions around the ease with which TCP attacks can be performed are unfounded
 - Blind hijacking is nearly impossible assuming good pseudorandom ISNs
 - Even "easy attacks" (TCP Resource Exhaustion) against port 179 are non-trivial against tight implementations and have minimal impact compared to other DoS attacks
- Why bother with lower layer attacks (ARP, TCP) against BGP when you can own the box?

More Conclusions

- Encourage your vendors to to test their BGP implementations and do your own security testing
 - These tests should be repeatable using this document and the BGP Attack Tree
- Implement BGP BCPs, especially admin ports!
- Liberally use clue-stick next time someone says "BGP is totally insecure!"
 - Security isn't an all or nothing proposition
 - soBGP[21] and S-BGP improve security, but...
 - New implementations, new bugs
 - Needs to go through the IETF process

What next?

- Generate more test-cases (more on BGP update and other message types)
- Test more platforms!
 - Need vendors, users, and independent researchers to repeat and extend tests we've outlined here
 - Based on "Active ISP Survey" there are more BGP implementations that need to be tested

References

- [1] S-BGP <u>http://www.net-tech.bbn.com/sbgp/sbgp-index.html</u>
- [2] BGP Security Vulnerabilities Analysis, Murphy, 2003http://www.ietf.org/internet-drafts/draft-murphy-bgp-vuln-02.txt
- [3] BGP Security Protections, Murphy, 2002 http://www.ietf.org/internet-drafts/draft-murphy-bgp-protect-01.txt
- [4] Expert: Router holes threaten Net http://zdnet.com.com/2100-1105-990608.html
- [5] Cisco Router Security Recommendation Guides http://www.nsa.gov/snac/cisco/guides/cis-2.pdf
- [6] An Attack Tree for the Border Gateway Protocol, Convery et. al., 2003- <u>http://trinux.sourceforge.net/draft-convery-bgpattack-</u> <u>00.txt</u>

References (cont.)

[7] Hping - http://www.hping.org

[8] Nemesis - http://www.packetfactory.net/projects/nemesis/

[9] Netcat -

http://www.atstake.com/research/tools/network_utilities/

[10] Naptha - <u>http://packetstormsecurity.nl/0101-exploits/</u>

[11] Dsniff - http://www.monkey.org/~dugsong/dsniff/

- [12] The Problem with Random Increments, Newsham, 2001 http://comsec.theclerk.com/CISSP/The%20Problem%20with%2 0Random%20Increments_4_30_01_final.PDF
- [13] Strange Attractors and TCP/IP Sequence Number Analysis, Zalewski, 2001 -

http://razor.bindview.com/publish/papers/tcpseq.html

References (cont.)

- [14] Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing, RFC 2827, BCP 38, Ferguson and Senie, 2000 - <u>http://www.ietf.org/rfc/rfc2827.txt</u>
- [15] Protection of BGP Sessions via the TCP MD5 Signature Option, RFC2385, Heffernan, 1998 - <u>http://www.ietf.org/rfc/rfc2385.txt</u>

[16] John the Ripper - http://www.openwall.com/john/

- [17] Security Requirements for Keys used with the TCP MD5 Signature Option, Leech, 2002 - <u>http://www.ietf.org/internet-</u> <u>drafts/draft-ietf-idr-md5-keys-00.txt</u>
- [18] The BGP TTL Security Hack (BTSH), Gill et. al., 2002 http://www.ietf.org/internet-drafts/draft-gill-btsh-01.txt

[19] Secure BGP Template, Thomas, v.2.9 http://www.cymru.com/Documents/secure-bgp-template.html

References (cont.)

[20] Hacking Layer 2: Fun with Ethernet Switches, Convery, 2002 http://www.blackhat.com/presentations/bh-usa-02/bh-us-02convery-switches.pdf

[21] Extensions to BGP to Support Secure Origin BGP (soBGP), Ng, 2002 - <u>http://www.ietf.org/internet-drafts/draft-ng-sobgp-bgp-</u> <u>extensions-00.txt</u>

[22] The Naptha DoS Vulnerabilities, Keyes.

http://razor.bindview.com/publish/advisories/adv_NAPTHA.html

[23] Cisco ISP Essentials, Greene - <u>ftp://ftp-eng/cons/isp/essentials</u>

