Understanding BGP Misconfiguration

Research question

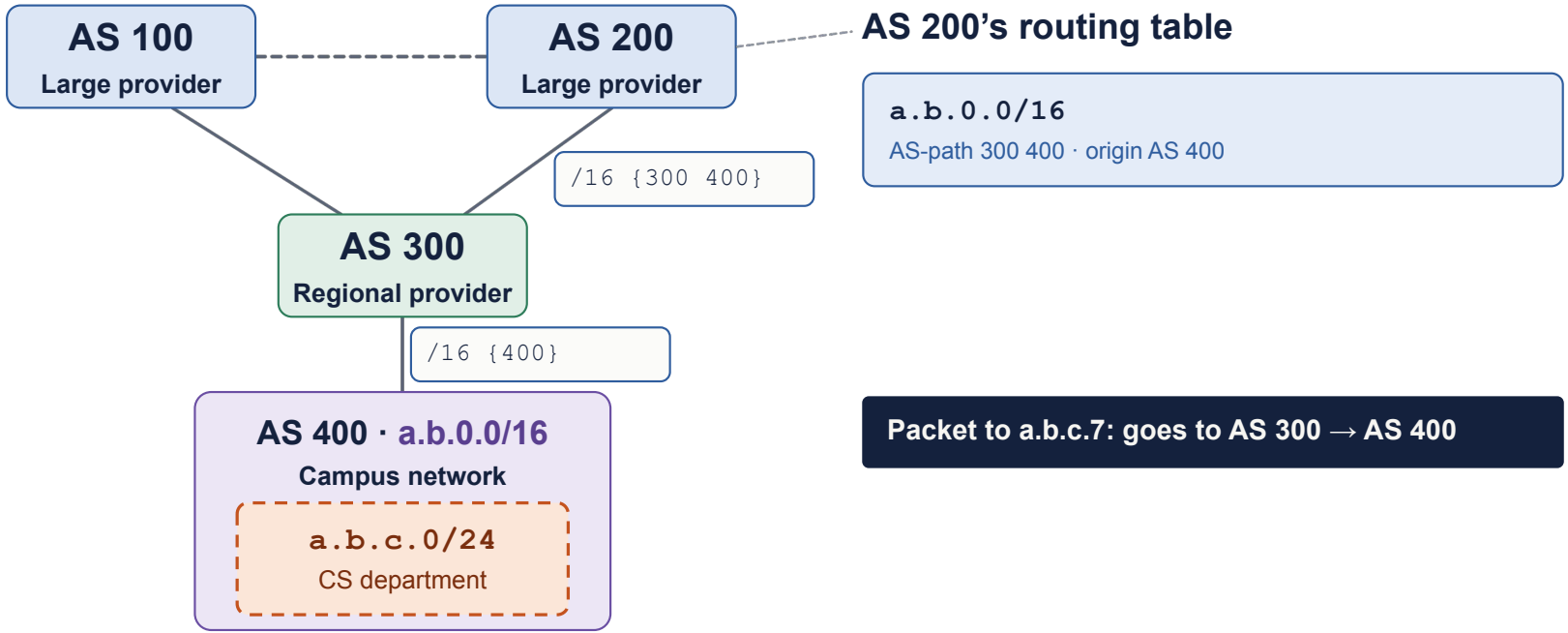
BGP misconfiguration happens: AS 7007 (1997) announced routes to most of the Internet and disrupted connectivity for hours.

How often does it happen?

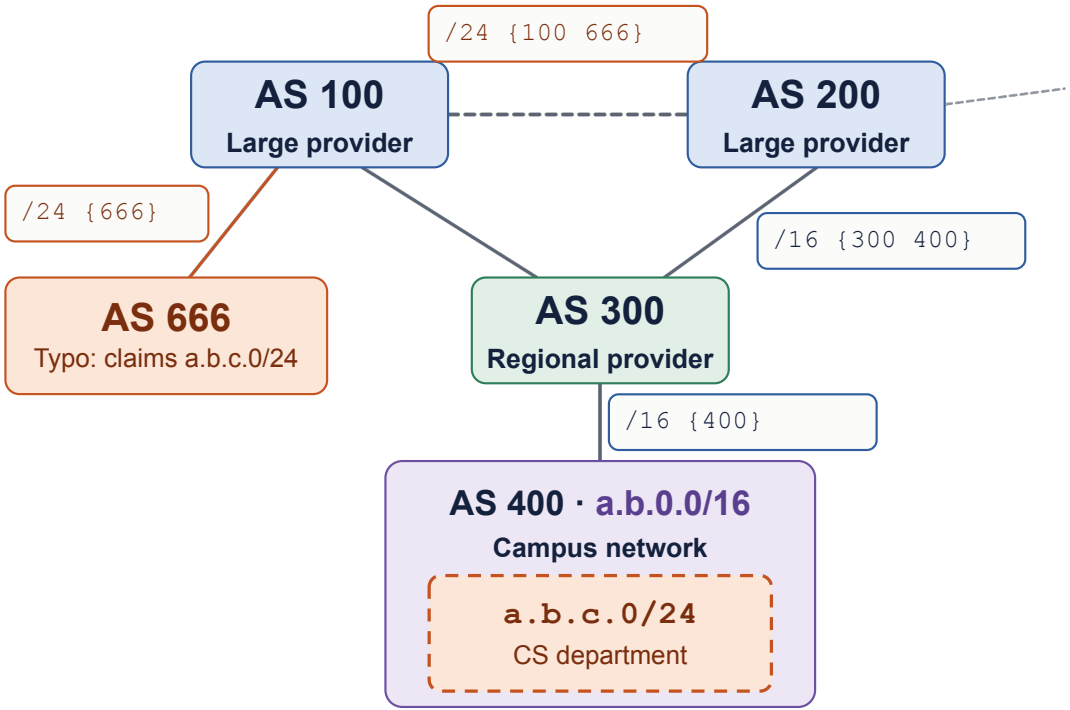
- Frequency
- Impact
- Causes

Estimate the lower-bounds of misconfiguration based on globally-visible information

Background: BGP announcements



Background: How configuration can go wrong



AS 200's routing table

a.b.0.0/16
AS-path 300 400 · origin AS 400

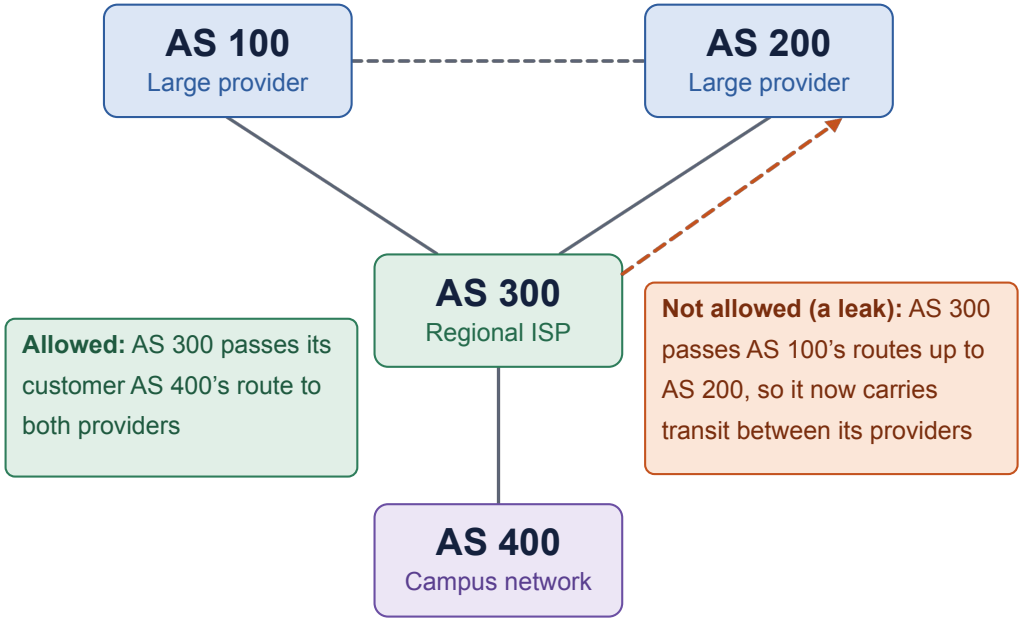
a.b.c.0/24
AS-path 100 666 · origin AS 666

Packet to a.b.c.7: goes to AS 100 → AS 666

Background: How configuration can go wrong

Learned from	To customers	To peers	To providers
A customer	Yes	Yes	Yes
A peer	Yes	No	No
A provider	Yes	No	No

- Routes learned from a customer are told to everyone, because you're paid to make your customers reachable
- Routes learned from a peer or a provider are told only to your customers. Otherwise you'd be giving away free transit.
- Valid AS-paths are **valley-free**



Background: How configuration can go wrong

AS 300's router (Cisco-style, simplified)

```
router bgp 300
 neighbor 1.1.1.1 remote-as 100
 neighbor 2.2.2.2 remote-as 200
 neighbor 2.2.2.2 route-map TO-AS200 out ①
 aggregate-address d.e.0.0 255.255.0.0 summary-only ②
 redistribute ospf 1 route-map OSPF2BGP ③
!
route-map TO-AS200 permit 10
 match ip address prefix-list CUSTOMERS
!
ip prefix-list CUSTOMERS permit a.b.0.0/16
```

① Export filter
Wrong or missing, and routes leak to a neighbor: an export misconfiguration.

② summary-only
Forget this keyword and every smaller prefix inside the /16 is announced too.

③ redistribute
Without a tight route-map, the whole internal table pours into BGP (AS 7007, 1997).

Background: How configuration can go wrong

Origin misconfiguration: A route appears that shouldn't exist

- Self-deaggregation
- Related origin
- Foreign origin

Export misconfiguration: A real route goes where policy forbids ("valley" in the path)

- provider → AS → provider
- provider → AS → peer
- peer → AS → provider
- peer → AS → peer

Methodology

21 days: 26 December 2001–15 January 2002. 23 BGP peers in 19 ASes.



Methodology

45% of new routes last less than one day; 30% last more than seven days.

The study investigates changes lasting less than a day, then asks operators whether they were intentional.

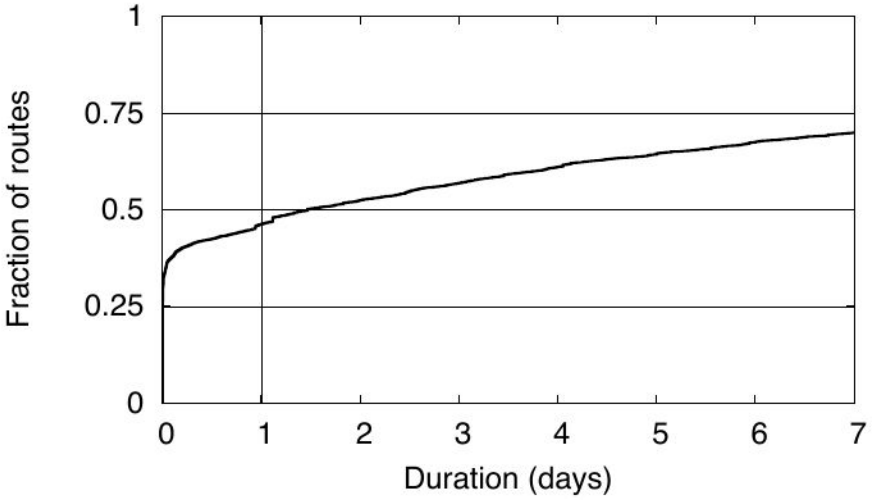


Figure 3: Fraction of new routes lasting $\leq x$ days.

Origin misconfiguration

Comparing against previous day's data to find new prefix never seen before / existing prefix announced by a new AS

Detect those exist for <1 day

	Old route	New route
Self deagg-regation	a.b.0.0/16 X Y Z	a.b.c.0/24 X Y Z
Related origin	a.b.0.0/16 X Y Z	a.b.0.0/16 X Y a.b.0.0/16 X Y Z O a.b.c.0/24 X Y a.b.c.0/24 X Y Z O
Foreign origin	a.b.0.0/16 X Y Z	a.b.0.0/16 X Y O a.b.c.0/24 X Y O e.f.g.h/i X Y O

Table 1: Classification of origin misconfigurations.

Announcement of smaller piece of its own block. Someone forgot to aggregate

Different origin appearing on old route's AS-path (e.g., provider leaks customer's prefix)

Unrelated AS announces the prefix (hijack), usually from typos

Export misconfiguration

Infer relationships between ASes based on Gao's algorithm

- Valid paths are valley-free
- A path has at most one peer link
- ASes with may neighbors tend to be providers

Route export	Export policy
Customer → Provider	Only routes received from customers and siblings
Peer → Peer	Only routes received from customers and siblings
Provider → Customer	All routes
Sibling → Sibling	All routes

Table 2: Export policies for common commercial relationships. Providers provide transit to customers; peers exchange only traffic that is sourced and sinked by them, their customers or their siblings; and siblings provide mutual transit.

Export	Policy violation
Provider→AS→Provider	Route exported to provider was imported from a provider
Provider→AS→Peer	Route exported to peer was imported from a provider
Peer→AS→Provider	Route exported to provider was imported from a peer
Peer→AS→Peer	Route exported to peer was imported from a peer

Table 3: Classification of route export misconfigurations.

Validation by asking operators and testing connectivity

For each incident, authors sent emails to ASes to confirm the incidents

- 30%: bounced due to contacts were stale
- Useful answers covered ~30% of incidents

Independently, test connectivity from multiple vantage points

Results

200–1,200 prefixes hit per day, about 3 in 4 new routes are mistakes, and operators confirmed nearly all flagged suspects

Self-deaggregation dominates by prefix count because single incidents can leak hundreds of prefixes

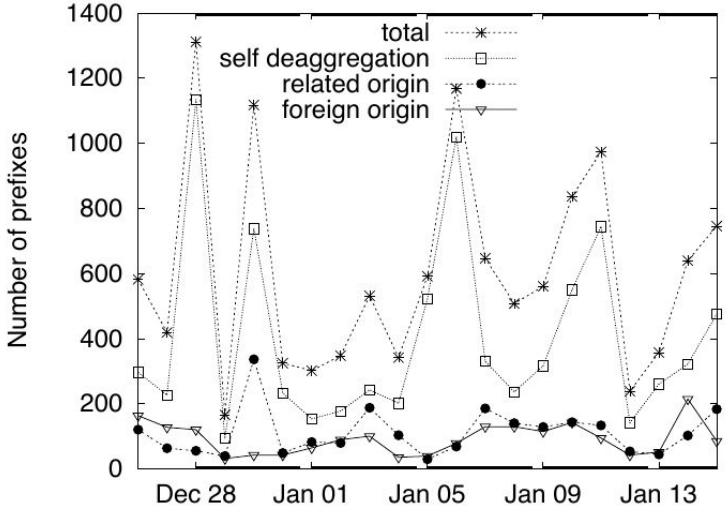


Figure 4: Probable origin misconfigurations per day.

Results

Only 4% of misconfigured prefixes (13% of incidents) break connectivity, and most errors are fixed within an hour

Type	Prefixes (top) Incidents (bottom)	Classified (% of total)	Misconfigurations All (% of classified)	 Connectivity (% of misconfigs)	Non-misconfigurations (% of classified)
Self deaggregation	8424	5598 (66%)	5519 (99%)	74 (01%)	79 (01%)
	1648	616 (37%)	563 (91%)	13 (02%)	53 (09%)
Related Origin	2341	1153 (49%)	1068 (93%)	54 (05%)	85 (07%)
	969	229 (24%)	188 (82%)	25 (13%)	41 (18%)
Foreign origin	1951	642 (33%)	535 (83%)	142 (27%)	107 (17%)
	1131	257 (23%)	194 (75%)	85 (44%)	63 (25%)
Total	12716	7393 (58%)	7122 (96%)	270 (04%)	271 (04%)
	3748	1102 (29%)	945 (86%)	123 (13%)	157 (14%)

Table 4: Results for origin misconfiguration. The classified column lists the number classified using email replies. Out of those, the all subcolumn specifies the number of misconfigurations, the connectivity subcolumn lists misconfigurations that disrupted connectivity, and the non-misconfigurations column specifies the number that were not misconfigurations (i.e. false positives in our analysis).

Results

For origin, mistakes (errors in plan) and slips (errors in execution for correct plan) were roughly equally responsible. For export, slips are much bigger cause.

Misconfigurations	*Prefixes*		*Incidents*		*Type*
	Total	Connectivity	Total	Connectivity	
Initialization bug	1580 (22%)	0 (00%)	43 (05%)	0 (00%)	mistake
Reliance on upstream filtering	977 (14%)	0 (00%)	431 (46%)	0 (00%)	mistake
Old configuration	72 (01%)	28 (39%)	36 (04%)	20 (56%)	mistake
Redistribution	2294 (32%)	1 (00%)	43 (05%)	1 (02%)	slip
Community	99 (01%)	2 (02%)	28 (03%)	2 (07%)	slip
Hijack	101 (01%)	101 (100%)	54 (06%)	54 (100%)	slip
Forgotten filter	53 (01%)	1 (02%)	13 (01%)	1 (08%)	slip
Incorrect summary	26 (00%)	0 (00%)	17 (02%)	0 (00%)	slip
Unknown configuration error	1053 (15%)	39 (04%)	90 (10%)	12 (13%)	slip
Miscellaneous	88 (01%)	16 (18%)	38 (04%)	10 (26%)	
Unclassified	779 (11%)	82 (11%)	152 (16%)	23 (15%)	
Non-misconfigurations	*Prefixes*		*Incidents*		
Failure	91 (34%)		50 (32%)		
Testing	66 (24%)		44 (28%)		
Migration	51 (19%)		26 (17%)		
Load balancing	22 (08%)		20 (13%)		
Miscellaneous	11 (04%)		7 (04%)		
Unclassified	30 (11%)		10 (06%)		

Misconfigurations	Paths	Incidents	Type
Prefix based config	98 (08%)	46 (22%)	mistake
Old configuration	20 (02%)	9 (04%)	mistake
Initialization bug	18 (01%)	9 (04%)	mistake
Bad ACL or route map	445 (34%)	8 (04%)	slip
Typo	153 (12%)	13 (06%)	slip
Forgotten filter	109 (08%)	15 (07%)	slip
Community	69 (05%)	37 (18%)	slip
Unknown config error	193 (15%)	14 (07%)	slip
Miscellaneous	22 (02%)	5 (02%)	
Unclassified	162 (13%)	54 (26%)	
Non-misconfigurations	Paths	Incidents	
Backup Arrangement	22 (47%)	13 (38%)	
Special Arrangement	13 (28%)	11 (32%)	
Failure	9 (19%)	7 (21%)	
Unclassified	3 (06%)	3 (09%)	

Results

Prefix based config (export misconfiguration):

A exports C's prefixes to P1 and P2. After link A–C fails, A may learn C's route through P1.

A prefix-only filter can then export that route to P2, unintentionally providing transit between providers.

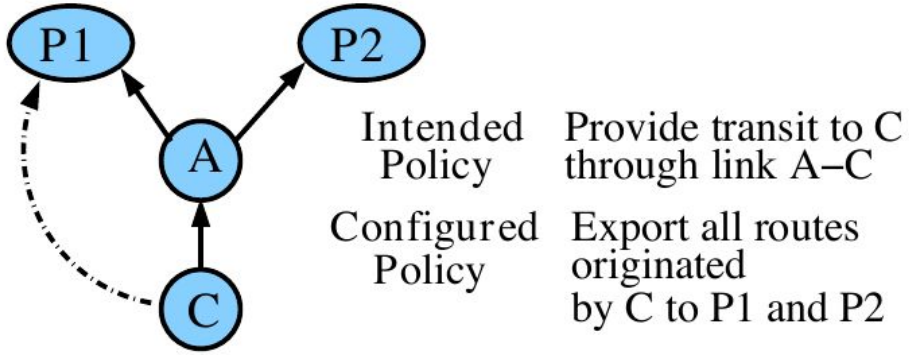


Figure 12: Prefix based configuration. The misconfiguration is uncovered when the link A-C fails.

How to fix it

Making systems need little manual configuration

1. User interface design
2. High-level languages and checking
3. Database consistency and registries
4. Protocol extensions

Discussion

1. This paper is published in 2002. How much of them are solved in 2026? What are still relevant?
2. What's the security implications? Can anyone maliciously misconfigure BGP for their profit?