AS Path Filtering Configuration Example

- AS Path filtering allows you to filter routes in incoming or outgoing updates. You permit or deny prefixes based on AS Path information.

```
ip as-path access-list 1 deny _65001_
ip as-path access-list 1 permit .*
```

- There is a default deny all at the end of the AS Path Access List.

```
router bgp 65000
 neighbor 1.1.1.1 filter-list 1 in
```

AS Path Filtering Configuration with Route Map

- A Route Map can set attributes on routes as well as filtering them

```
ip as-path access-list 2 permit _65002_

route-map NEIGHBOR_2 permit 10
 match as-path 2
 set local-preference 200

router bgp 65000
 neighbor 2.2.2.2 route-map NEIGHBOR_2 in
```

Regular Expressions

- When AS Path filtering, the ASs are referenced with regular expressions.
- A regular expression (RegEx) is a sequence of characters that specifies a search pattern in text (printable ASCII characters including numbers).
- They can be used in many different computing functions to match on (and possibly take an action on such as to replace) particular patterns in text.
- Modifiers (wildcards) can be used to provide very flexible matching.
- RegEx syntax can vary slightly between platforms.

Common Regular Expression Modifiers

.	matches any single character, including a space. eg 91. matches 917 or 91b or ’91 ’ but not ’91’
*	preceding character or group can be found zero or more times, eg 7* matches 7 or 7777 or ‘’
.*	matches any string of any length, including a blank string
+	preceding character or group can be found one or more times, eg 7+ matches 7 or 7777 but not ‘’
.+	matches any string as long as it has at least one character
?	preceding character or group can be found zero or one times, eg 917? matches 91 or 917 but not 91777. (Use Ctrl-V)
^	matches on the start of a string, eg ^9.* matches 9abc1 or 91 or 9596 but not 567 or 8yz92
$	matches on the end of a string, eg .*1$ matches 9abc1 or 91 but not 567 or 81yz2
_	matches any delimiter (beginning, end, space, tab, comma), eg 4_5 matches ‘4 5’
()	groups part of a string, and assigns it a number (see next slide for example). (abc) matches abc
|	logical OR, eg p(abc|xyz)q matches pabcq OR pxyzq
[]	matches single character inside brackets, eg p[abc]q matches ‘paq’ or ‘pbq’ or ‘pcq’. It does not match ‘pabcq’
[^]	Matches any single character except what’s inside brackets, eg p[^ab]q matches pzq or p1q etc. but not paq or pbq
\	remove special meaning of following character, eg \. means match literally . (a dot, not any character)

RegEx Example - Telephony

/(^9)(.+)/ /\2/

FLACKBOX
www.flackbox.com

- Staff in the office dial 9 for an outside line
- The 9 must be stripped off before sending the digits to the Telco. For example if 98495009 is dialled then 8495009 should be sent to the Telco
- The string searched for is inside the 1st set of //
- The string to replace it with is inside the 2nd set of //
- (^9) looks for a string beginning with 9, and groups it with group no. 1
- (.+) matches on whatever comes after the 9, and groups it with group no. 2
- \2 copies group no. 2 as the replacement string

Regular Expressions

- Regular Expressions (other than ^ and $) will match on any part of a string, you do not need to specify the entire string
- _65001_ will match any AS path that passes through AS 65001. You do not need to use .*_65001_.*
- In the previous telephony example, we could have used /^9/ //
- This will strip off the leading 9, leaving the remaining digits. Dialing 98495009 would result in 8495009
- /9/ // would be a misconfiguration and remove all 9s from the number, eg dialing 98495009 would result in 84500
- Using ^ and $ is a common way to avoid this problem

AS Path Regular Expression Examples

^$	Networks originated in local AS. (Blank string)
^65001$	Networks originated in neighbor AS 65001
_65001$	Networks originated in AS 65001
_65001_	Networks with AS 65001 in AS Path
^65001_	Networks through neighbor AS 65001
^65001_65002$	Networks with AS path of exactly 65001_65002
^65001_?.*_65002$	Networks originated in AS 65002, through neighbor 65001, any (or no) path in between
^[0-9]+$	Networks originated in any neighbor. (AS Path is one AS long, any AS number)
.*	Matches everything

Verification – show ip bgp regex

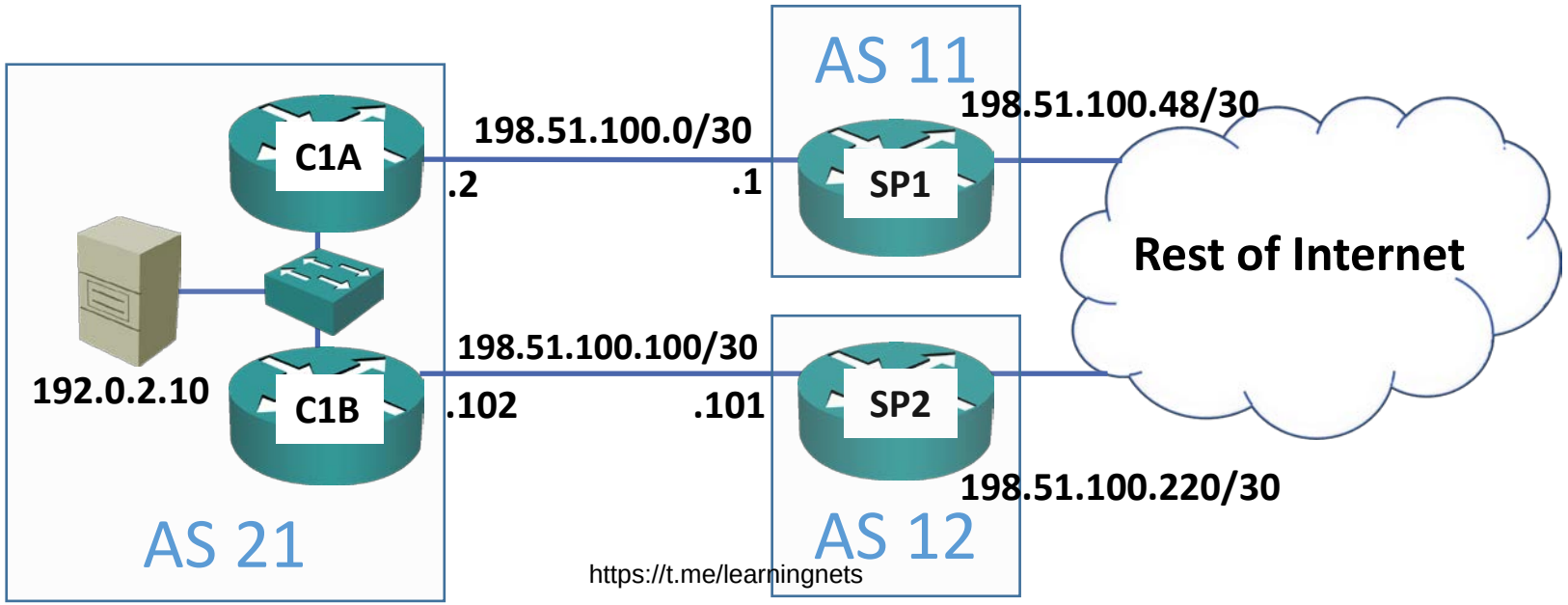
- Shows only routes which match the Regular Expression

```
C2#show ip bgp regexp _65001_

     Network          Next Hop            Metric LocPrf Weight Path
 *    192.0.2.0        198.51.100.9                           0 65001 65011 i
 *>                    198.51.100.5                           0 65001 65011 i
 *                     192.168.1.1                            0 65003 65002 65001 65011 i
 *    198.51.100.4/30  192.168.1.1                            0 65003 65002 65001 i
 *>                    198.51.100.9                           0 65001 i
```

Verification – show ip as-path-access-list

```
C1A#show ip as-path-access-list
AS path access list 1
    permit ^11$
```



Verification – show ip bgp filter-list

- Shows only routes which match the Filter List

FLACKBOX
www.flackbox.com

```
C1A#show ip bgp filter-list 1

 Network          Next Hop            Metric LocPrf Weight Path
*> 198.51.100.48  198.51.100.1                           0 11 i
```