Matthew Williams

From: Maximo Alves [maximo@ripe.net]
Sent: 22 April 2003 14:06
To: ris@ripe.net
Subject: myAS Design - Scan&Match(last specs)

The scan&match functions are now accomplished remotely on the RRC's by db_insert and locally by scan&match.
Thus the requirements and specifications are divided.

DB_INSERT:

This is more or less what me and James discussed 2 weeks ago. James, please have a look to check if everything is fine!

Requirements:

- Load User config from local file during startup. Subsequently load all the times it changes. Change flaged by time of last modification
- match raw BGP data against User configuration
- If match is positive(thus alarm event is trigered) set myASflag attribute on Updates table to:
  * 1 (in case match is prefix based)
  * 2 (in case match is aspath based)
  * 3 (in case both matches are positive)
- If match is NOT positive set myASflag to:
  * 0
- Only flag Advertisement messages

Especifications:

- Data structure for prefix based alarm:
  - binary tree
  - each node is a C structure containing the current elements plus:
    - flag: 0 don't flag
            1 flag if origin AS in advertisement <> asuser
            2 more than one asuser have configured this prefix so flag all the time
    - asuser: as number

- Datastructure for aspath based alarm:
  - use array filled with regular expressions

- Rules to trigger prefix based alarm:
  - given that prefixA is advertised by asXX
  - find prefixA position in binary tree
  - depending on flag value:
    - if flag=0 do nothing
    - if flag=1 is asXX == asuser
      - if yes write 1 to myASflag on Updates
    - if flag=2 write 1 to myASflag on Updates

- Rules to trigger aspath based alarm:
  - given a ASPATH
  - try it with all regular expressions
  - if one or more return true:
    - write 2 to myASflag on Updates

- The config file has the following format:

```
PREFIX ORIGIN prefix asuser
ASPATH REGEX 'regular expression'

Obs: lines end with \n
     words are separated by \s
```

SCAN&MATCH:

Requirements:

- Load User config from local file during startup. Subsequently load all the times it changes. Changed flaged by time of last modification
- Query Updates table(one table for each RRC) to retrieve all records dating from the last n seconds where myASflag= (1 | 2 | 3) AND Type = Advertisement .
- Query Updates table(one table for each RRC) to retrieve all records dating from the last n seconds where Type = withdraw
- Create and/or update records in the Alarm DB. Tables: origin, transit, customer, prefix, peer, aspathbit
- Cash mechanism to reduce queries to RIS DB in the long term
- Signal in GUI when system is "out of order"

Especifications:

will follow later.

Matthew Williams

From: Maximo Alves [maximo@ripe.net]
Sent: 09 May 2003 10:46
To: ris@ripe.net
Subject: [myAS Design - Aggregator]

REQUIREMENTS:

- reads alarm events from all tables(origin, transit and customer) aggregating them im single mail messages per User
- send reset alarm messages when conflicting announcements are 100% clear(or another percentage)
- Origin alarms are aggregated by prefix(the same found in the User's configuration)
- Transit and Customer alarms are aggregated by aspathbit
- deletes alarm events from tables when time to live expires
- cash mechanism in order to keep track of messages that were sent
- user setting to indicate aggregation interval(every 5 mins or n times 5 mins).

- Alarm Message example:

Dear myAS User,
Account: 1897

The following announcements conflicted with your configuration and have triggered the alarm events bellow. You can find more information looking at www.ris.ripe.net/myas/

ORIGIN alarm:

1 - Configured prefix: 193.126.0.0/16
Alarm details:

Prefix	Peer	Announcing AS	Time it fist occurred	RRC
193.126.0.0/16	212.13.32.0	12305	yyyyddmm:hhmmss	0(A'dam)
193.126.0.0/16	212.13.32.0	701	yyyyddmm:hhmmss	1(London)

2 - Configured prefix: 193.127.0.0/16
Alarm details:

Prefix	Peer	Announcing AS	Time it fist occurred	RRC
193.127.0.0/16	212.13.32.0	12305	yyyyddmm:hhmmss	2(...)
193.127.4.0/24	212.13.33.0	701	yyyyddmm:hhmmss	1(...)
193.127.4.0/24	212.13.33.0	700	yyyyddmm:hhmmss	1(...)

TRANSIT alarm:

1 - ASpathBit: 7901 1897
Alarm details:

Prefix	Peer	Time it fist occurred	RRC
193.127.0.0/16	212.13.32.0	yyyyddmm:hhmmss	2(...)
193.127.4.0/24	212.13.33.0	yyyyddmm:hhmmss	1(...)

2 - ASpathBit: 1111 x x x 1897
Alarm details:

Prefix	Peer	Time it fist occurred	RRC
193.127.2.0/24	212.13.32.0	yyyyddmm:hhmmss	2(...)
193.127.4.0/24	212.13.33.0	yyyyddmm:hhmmss	1(...)

CUSTOMER alarm:

1 - ASpathBit: 1897 5555
Alarm details:

```
  Prefix       -    Peer        - Time it fist occurred - RRC
193.127.0.0/16 - 212.13.32.0    -    yyyyddmm:hhmmss    - 2( ... )
193.127.4.0/24 - 212.13.33.0    -    yyyyddmm:hhmmss    - 1( ... )

Best regards,

myAS.
```

- Reset Alarm Message example:

```
Dear myAS User,
Account: 1897

Some announcements that conflicted with your configurations were
reseted by withdraw messages. See the list bellow. You can find more information
looking at www.ris.ripe.net/myas/

ORIGIN alarm:
193.126.0.0/16
193.127.0.0/16

CUSTOMER alarm:
1897 5555

Best regards,

myAS.
```

PROCESS SPECIFICATION:

Data structure:

%usercash:
 It's loaded during initialization and saved to file regularly
 Key: asuser
 Value: reference to %sent

%sent:
 Store alarm events that were aggregate in messages(thus sent)
 Key: prefix or aspathbit
 Value: 1

%off:
 Key: prefix or aspathbit
 Value: 1

%on:
 Key: prefix or aspathbit
 Value: 1

Algorithm:

- Initialization
  - Read alarm config
  - Read system config (alarm types, default dir, MySQL domain, etc)
  - Load from file %usercash
- while(true)
  - Foreach User
    - Foreach Alarm type

```
    - Select from DB (SQL statement 22, 23 and 24) and put into @array
    - Foreach $line (@array)
       - If(time to live has expired)
         - add alarm id to @delbuffer
       - $key = prefix | aspathbit
       - Next If($sent{$key})
       - If(Holddowntime | Holddownevent) ?????BASELINE?????
         - add $line to @sendbuffer
         - $sent{$key} = 1
       - If(status eq 'a') $on{$key} = 1
       - If(status eq 'o') $off{$key} = 1
    - End foreach $line
  - End foreach Alarm type
  - Foreach $line (@sendbuffer)
    - aggregate_alarm_message ($line)
  - Send Alarm messages
  - Foreach $key (%on)
    - If $key exists in %off aggregate_reset_messages($key)
  - Send Reset Alarm messages
  - Open connection to MySQL server
  - Foreach $line(@delbuffer)
   - del record (SQL statement 25, 26, 27)
   - add record to log Table
  - Close connection to MySQL server
  - clean %on, %off
- End foreach User
- save %usercash to file
- sleep()
```

Matthew Williams

From: Maximo Alves [maximo@ripe.net]
Sent: 09 May 2003 11:05
To: ris@ripe.net
Subject: [myAS Design - retromatch]

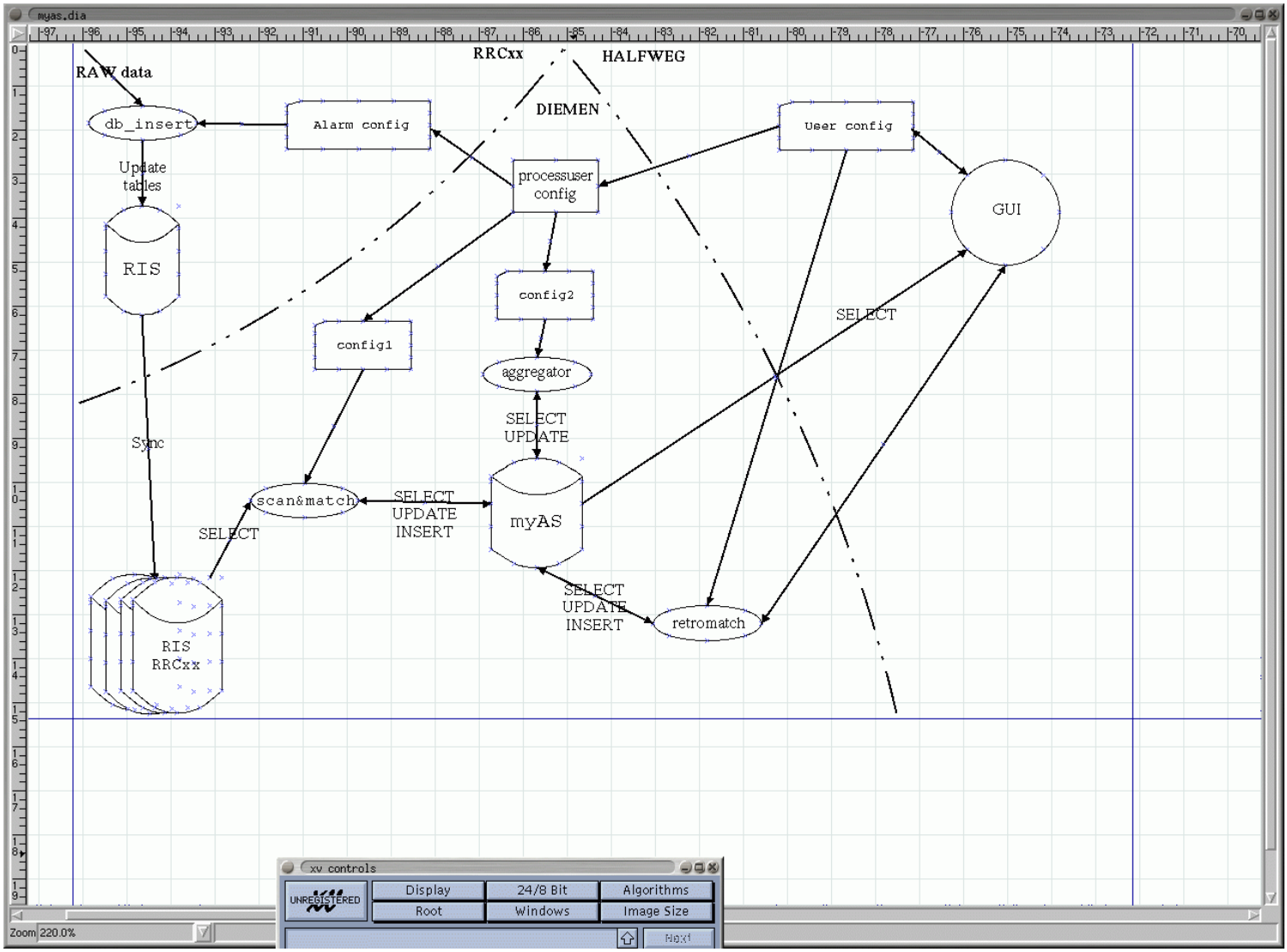
It implements the retroactive alarm generation function.

Requirements:

- runs in the background with lower priority
- time period for alarm generation:
  * upper limit: current day
  * lower limit: 90 days before the upper limit(or whatever RIS stores)
  * User is free to choose any consecutive days inside this period
- starts alarm event generation only on demand
- get User input from CGI
- update temporary retroactive table in Alarm DB
- update GUI during alarm processing in order to
  keep User informed:
  * make user aware of estimated processing time
- internal cashing decrease, in time, queries to RIS DB.
- cash intermediate results and whenever possible always reuse them
  in order to decrease processing time:
  * usefull in case User changes a threshold that doesn't modify RIS DB
  flagging
- Only the final alarm state is stored in Alarm DB for GUI
  interaction.
- doesn't log

Specifications:

will follow later(want to talk to Rene first)


myas.dia
RAW data
RRCxx
HALFWEG
DIEMEN
db_insert
Alarm config
User config
Update
tables
processuser
config
GUI
RIS
config2
config1
aggregator
SELECT
SELECT
UPDATE
Sync
scan&match
SELECT
UPDATE
INSERT
myAS
SELECT
SELECT
UPDATE
INSERT
retromatch
RIS
RRCxx
xv controls
Display
24/8 Bit
Algorithms
Root
Windows
Image Size
Zoom 220.0%

Sheet1

		myAS Database Structure	
origin Table	**data type**	**description**	**indexed**
Id	INT Unsig	Auto-Incrementing, Unique	no
myas	Medium INT Unsig	User account AS number	x
config_prefix_id	INT Unsig	prefix found in the User Origin	no
		configuration(reference to prefix table)	
prefix_id	INT Unsig	announced prefix (reference to prefix table)	no
peer_id	INT Unsig	bgp peer(reference to peer table)	no
offendingas	Medium INT Unsig	AS announcing conflicting prefix	no
occurrences	INT Unsig	number of times this particular alarm evet	no
		has happened	
status	CHAR(1)	a: active	no
		d:deleted	
		o: off	
rrc	Tiny INT Unsig	rrc from where the update originates	no
starttime	INT Unsig	first time alarm happened (unix timestamp)	no
endtime	INT Unsig	last time alarm happened (unix timestamp)	no
transit Table			
id	INT Unsig	Auto-Incrementing, Unique	no
myas	Medium INT Unsig	User account AS number	x
aspathbit_id	INT Unsig	aspath bit that triggered alarm event	no
prefix_id	INT Unsig	announced prefix (reference to prefix table)	no
peer_id	INT Unsig	bgp peer(reference to peer table)	no
occurrences	INT Unsig	number of times this particular alarm evet	no
		has happened	

status	CHAR(1)	a: active	no
		d:deleted	
		o: off	
rrc	Tiny INT Unsig	rrc from where the update originates	no
starttime	INT Unsig	first time alarm happened (unix timestamp)	no
endtime	INT Unsig	last time alarm happened (unix timestamp)	no
customer Table			
id	INT Unsig	Auto-Incrementing, Unique	no
myas	Medium INT Unsig	User account AS number	x
aspathbit_id	INT Unsig	aspath bit that triggered alarm event	no
prefix_id	INT Unsig	announced prefix (reference to prefix table)	no
peer_id	INT Unsig	bgp peer(reference to peer table)	no
occurrences	INT Unsig	number of times this particular alarm evet	no
		has happened	
status	CHAR(1)	a: active	no
		d:deleted	
		o: off	
rrc	Tiny INT Unsig	rrc from where the update originates	no
starttime	INT Unsig	first time alarm happened (unix timestamp)	no
endtime	INT Unsig	last time alarm happened (unix timestamp)	no
origin_log Table	**data type**	**description**	**indexed**
Id	INT Unsig	Auto-Incrementing, Unique	no
myas	Medium INT Unsig	User account AS number	x
config_prefix_id	INT Unsig	prefix found in the User Origin	no
		configuration(reference to prefix table)	
prefix_id	INT Unsig	announced prefix (reference to prefix table)	no
		bgp peer(reference to	

peer_id	INT Unsig	peer table)	no
offendingas	Medium INT Unsig	AS announcing conflicting prefix	no
occurrences	INT Unsig	number of times this particular alarm evet	no
		has happened	
rrc	Tiny INT Unsig	rrc from where the update originates	no
starttime	INT Unsig	first time alarm happened (unix timestamp)	no
endtime	INT Unsig	last time alarm happened (unix timestamp)	no
transit_log Table			
id	INT Unsig	Auto-Incrementing, Unique	no
myas	Medium INT Unsig	User account AS number	x
aspathbit_id	INT Unsig	aspath bit that triggered alarm event	no
prefix_id	INT Unsig	announced prefix (reference to prefix table)	no
peer_id	INT Unsig	bgp peer(reference to peer table)	no
occurrences	INT Unsig	number of times this particular alarm evet	no
		has happened	
rrc	Tiny INT Unsig	rrc from where the update originates	no
starttime	INT Unsig	first time alarm happened (unix timestamp)	no
endtime	INT Unsig	last time alarm happened (unix timestamp)	no
customer_log Table			
id	INT Unsig	Auto-Incrementing, Unique	no
myas	Medium INT Unsig	User account AS number	x
aspathbit_id	INT Unsig	aspath bit that triggered alarm event	no
prefix_id	INT Unsig	announced prefix (reference to prefix table)	no
peer_id	INT Unsig	bgp peer(reference to peer table)	no
occurrences	INT Unsig	number of times this particular alarm evet	no

		has happened	
rrc	Tiny INT Unsig	rrc from where the update originates	no
starttime	INT Unsig	first time alarm happened (unix timestamp)	no
endtime	INT Unsig	last time alarm happened (unix timestamp)	no
origin_retroactive Table	**data type**	**description**	**indexed**
Id	INT Unsig	Auto-Incrementing, Unique	no
myas	Medium INT Unsig	User account AS number	x
config_prefix_id	INT Unsig	prefix found in the User Origin	no
		configuration(reference to prefix table)	
prefix_id	INT Unsig	announced prefix (reference to prefix table)	no
peer_id	INT Unsig	bgp peer(reference to peer table)	no
offendingas	Medium INT Unsig	AS announcing conflicting prefix	no
occurrences	INT Unsig	number of times this particular alarm evet	no
		has happened	
status	CHAR(1)	a: active	no
		d:deleted	
		o: off	
rrc	Tiny INT Unsig	rrc from where the update originates	no
starttime	INT Unsig	first time alarm happened (unix timestamp)	no
endtime	INT Unsig	last time alarm happened (unix timestamp)	no
transit_retroactive Table			
id	INT Unsig	Auto-Incrementing, Unique	no
myas	Medium INT Unsig	User account AS number	x
aspathbit_id	INT Unsig	aspath bit that triggered alarm event	no
prefix_id	INT Unsig	announced prefix (reference to prefix table)	no
		bgp peer(reference to	

peer_id	INT Unsig	peer table)	no
occurrences	INT Unsig	number of times this particular alarm evet	no
		has happened	
status	CHAR(1)	a: active	no
		d:deleted	
		o: off	
rrc	Tiny INT Unsig	rrc from where the update originates	no
starttime	INT Unsig	first time alarm happened (unix timestamp)	no
endtime	INT Unsig	last time alarm happened (unix timestamp)	no
customer_retroactive Table			
id	INT Unsig	Auto-Incrementing, Unique	no
myas	Medium INT Unsig	User account AS number	x
aspathbit_id	INT Unsig	aspath bit that triggered alarm event	no
prefix_id	INT Unsig	announced prefix (reference to prefix table)	no
peer_id	INT Unsig	bgp peer(reference to peer table)	no
occurrences	INT Unsig	number of times this particular alarm evet	no
		has happened	
status	CHAR(1)	a: active	no
		d:deleted	
		o: off	
rrc	Tiny INT Unsig	rrc from where the update originates	no
starttime	INT Unsig	first time alarm happened (unix timestamp)	no
endtime	INT Unsig	last time alarm happened (unix timestamp)	no
prefix Table			
id	INT Unsig	Auto-Incrementing, Unique	
prefix	CHAR(42)	prefix string	
peer Table			
		Auto-Incrementing,	

id	INT Unsig	Unique
ipn	CHAR(42)	IP address
asnumber	Medium INT Unsig	

aspathbit Table

id	INT Unsig	Auto-Incrementing, Unique
regex	CHAR(255)	regular expression

SQL statements

GUI -> Alarm DB

1	SELECT id, Prefix.prefix, starttime, endtime FROM origin, prefix WHERE myas = 'asnumber' AND status <> 'd' GROUP BY prefix.prefix ORDER BY endtime
2	SELECT id, aspathbit.regex, starttime, endtime FROM transit, aspathbit WHERE myas = 'asnumber' AND status <> 'd' GROUP BY aspathbit.regex ORDER BY endtime
3	SELECT id, aspathbit.regex, starttime, endtime FROM customer, aspathbit WHERE myas = 'asnumber' AND status <> 'd' GROUP BY aspathbit.regex ORDER BY endtime
4	SELECT prefix.prefix, peer.ipn, origin.offendingas, origin.occurrences, origin.starttime, origin.endtime WHERE origin.myas = 'asnumber' AND origin.config_prefix_id = 'id' AND origin.status <> 'd' ORDER BY ...
5	SELECT prefix.prefix, peer.ipn, transit.occurrences, transit.starttime, transit.endtime WHERE transit.myas = 'asnumber' AND transit.aspathbit_id = 'id' AND transit.status <> 'd' ORDER BY ...
6	SELECT prefix.prefix, peer.ipn, customer.occurrences, customer.starttime, customer.endtime WHERE customer.myas = 'asnumber' AND customer.aspathbit_id = 'id' AND customer.status <> 'd' ORDER BY ...

Scan&Match -> RIS DB

7	SELECT prefix.prefix, aspath.aspath, peer.ipn, yyyymmdd.type FROM yyyymmdd, prefix, aspath, peer WHERE UTC > 'timestamp' AND peer.id = yyyymmdd.peer_id AND prefix.id = yyyymmdd.prefix_id
	AND aspath.id = yyyymmdd.aspath_id AND yyyymmdd.flag <> 0 AND ryyyymmdd.type <> 'S' (run it on all rrc db's)

Scan&Match -> Alarm DB

10	SELECT origin.id WHERE prefix.string = 'prefix' AND peer.string = 'prefix' AND origin.offendingas = 'asnumber'
11	SELECT transit.id WHERE prefix.string = 'prefix' AND peer.string = 'prefix' AND aspathbit.string = 'aspathbit'
12	SELECT customer.id WHERE prefix.string = 'prefix' AND peer.string = 'prefix' AND aspathbit.string = 'aspathbit'
13	UPDATE origin SET occurrences = occurrences + 1 WHERE id = 'id'
14	UPDATE transit SET occurrences = occurrences + 1 WHERE id = 'id'
	UPDATE customer SET occurrences = occurrences + 1 WHERE id =

15	'id'
16	UPDATE origin SET status = 'o' WHERE id = 'id'
17	UPDATE transit SET status = 'o' WHERE id = 'id'
18	UPDATE customer SET status = 'o' WHERE id = 'id'
19	INSERT INTO origin (... ...) VALUES (... ... ...)
20	INSERT INTO transit (... ...) VALUES (... ... ...)
21	INSERT INTO customer (... ...) VALUES (... ... ...)
	Agregator -> Alarm DB
22	SELECT * FROM origin WHERE myas = 'asnumber' AND endtime > 'time'
23	SELECT * FROM transit WHERE myas = 'asnumber' AND endtime > 'time'
24	SELECT * FROM customer WHERE myas = 'asnumber' AND endtime > 'time'
25	UPDATE origin SET status='d' WHERE id = 'id'
26	UPDATE transit SET status='d' WHERE id = 'id'
27	UPDATE customer SET status='d' WHERE id = 'id'