Atualizações de 2021 no TC IRR

Herbert Faleiros & Rubens Kühl

Era uma vez um IRR...

O TC foi criado em 2010 (como um projeto pessoal) com o objetivo de disponibilizar aos Sistemas Autônomos brasileiros um contato inicial descomplicado (wizard) e principalmente *gratuíto* com o IRR.

Até então, dos ASes nacionais com IRR, a maioria eram 'proxies' (cerca de 80%) e destes boa parte inválidos (por volta de 40%).

Ao final de 2019, desconsiderando os proxies, mais de 90% dos ASes brasileiros encontrados na DFZ estavam registrados no TC, portanto o objetivo inicial da base após quase uma década foi plenamente alcançado!

O TC contou com o apoio de diversas entidades e empresas durante sua existência, majoritariamente com a cessão de recursos computacionais.

Mudança 2021 #1: Hospedeiro e aumento de RAM

Após alguns anos na AWS, o TC ficou hospedado num provedor nacional (também pago) de VPS. Ambos forneciam máquinas com pouca RAM.

A partir de Abril de 2021, o TC passou a ser hospedado pela SoftDados numa VM agora com 32 GB de RAM e ótima performance de I/O

Mudança 2021 #2: Retorno e ampliação dos mirrors

Os poucos recursos da VM anterior tinham forçado a paralisação dos mirrors de outros IRR listados em http://irr.net

Com a nova VM, os mirrors foram retomados

Vários mirrors que antes eram de espelhamento total passaram a usar NRTM (Near Real Time Mirroring)

Mudança 2021 #3: IRRd 4.2

O aumento de RAM permitiu a migração para a nova versão do IRRd, que é escrita em Python e usa PostgreSQL e Redis

Inicialmente alterado em Maio de 2021 para 4.2.0b1, depois para 4.2.0 release, depois para 4.2.1

Primeiro IRR de produção a migrar para 4.2

IRRs, versões e número de sistemas autônomos

IRR	ASNs	versão
RIPE	37306	RIPE DB 1.101
APNIC	17737	APNIC WS 1.88.15
RADB	8813	IRRd 3.0.8
TC	3432	IRRd 4.2.1
ARIN	2223	IRRd 4.1.7
RIPE-NONAUTH	2163	RIPE DB 1.101
AFRINIC	2062	
IDNIC	1719	IRRd 4.1.0
ALTDB	1411	IRRd 2.3.10

IRR	ASNs	versão
LACNIC	1039	IRRd 4.1.0
ARIN-NONAUTH	938	IRRd 4.1.7
WCGDB	773	RIPE DB 3.0.0
NTTCOM	548	IRRd 4.1.8
JPIRR	425	IRRd 2.3.10
LEVEL3	299	RIPE DB 3.0.0
CANARIE	177	RIPE DB 1.98
BELL	105	IRRd 2.3.10
BBOI	56	IRRd 3.0.9rc2

Impacto #1 do IRRd 4.2: Checagem RPSL estrita

A versão de IRRd antes usada no TC, 3.0.9rc2, era menos estrita na checagem dos objetos (RPSL).

AS-SETs agora são necessariamente hierárquicos da forma ASnnnn:AS-nomedoASset, para garantir unicidade e relação com um maintainer

Contatos agora precisam ter uma referência person ou role

Notify precisa ser um e-mail, não pode ser um nic-handle

Essas regras já existiam na RPSL, mas não eram verificadas

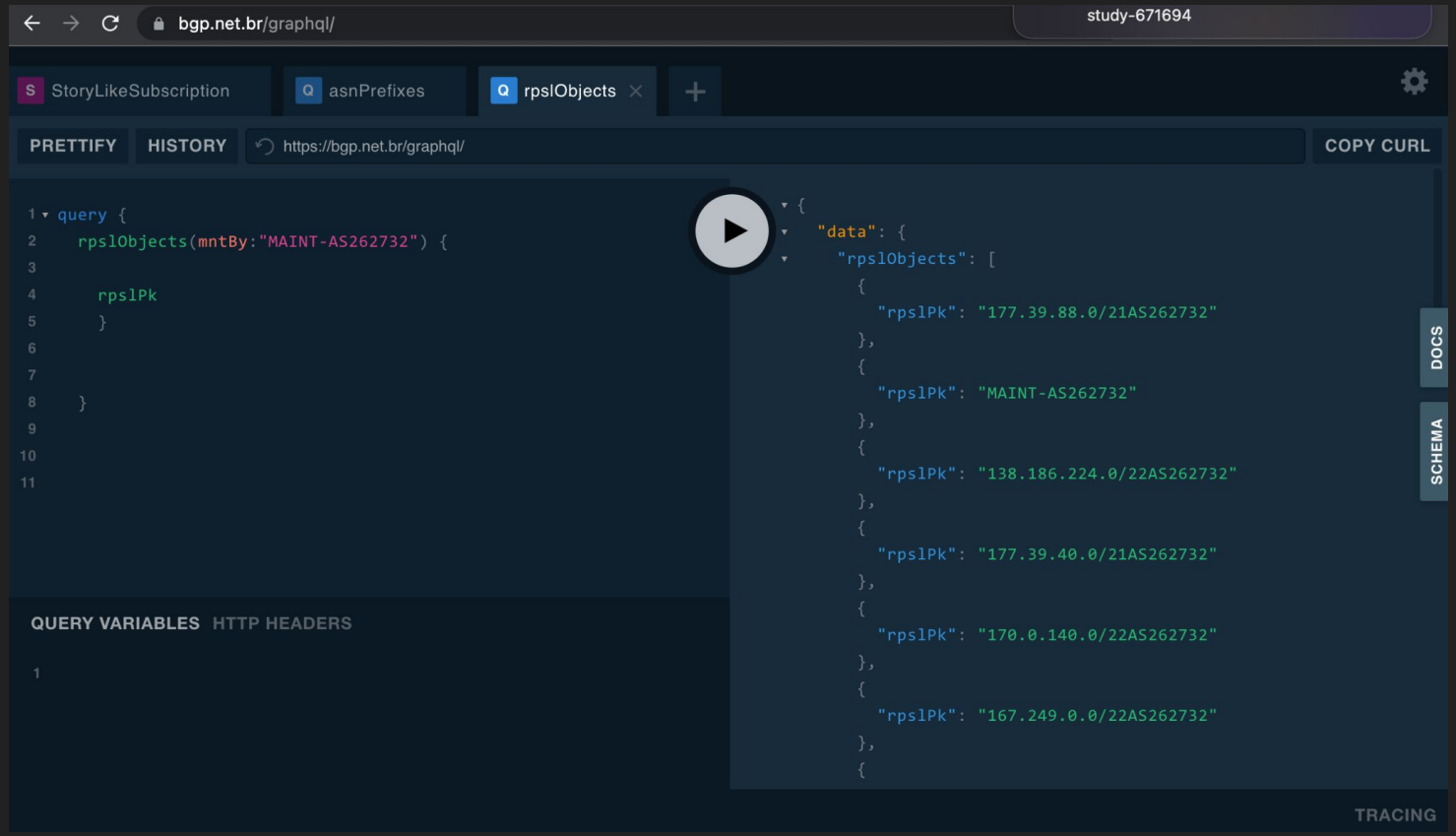
Impacto #2 do IRRd 4.2: APIs

Com o upgrade para 4.2, o TC passou a disponibilizar Web APIs. Elas incluem:

- WHOIS
- Submit
- GraphQL

Os scripts novos do TC já foram escritos para uso dessas APIs, e confirmaram seu bom funcionamento

GraphQL: Uso programático ou visual



Impacto #3 do IRRd 4.2: RPKI

Com o upgrade para 4.2, o TC passou a fazer checagem RPKI nos objetos próprios do TC (não nos mirrors)

Essa checagem é um dos maiores consumidores de RAM atualmente, apesar da redução entre as versões 4.2.0b1 e a 4.2.0 release

Apenas objetos RPKI inválidos são removidos, não os desconhecidos

A causa usual de remoções por RPKI são os próprios maintainers tentando adicionar mais específicos do que a política de RPKI daquele bloco permite

Impacto #4 do IRRd 4.2: Scope Filter

Com o upgrade para 4.2, o TC passou a usar o recurso scope filter nos objetos próprios do TC e nos mirrors

Ele descarta automaticamente IPs e ASNs inválidos

Há nítido overhead no mirror de bases grandes como o RADB e as dos RIRs

Como os objetos próprios do TC já são confrontados com as alocações do Registro.br, a efetiva utilidade é questionável

Pode vir a ser desligado, talvez seja mais útil para construção automatizada de filtros

Impacto #5 do IRRd 4.2: Omissão do hash da senha

Na versão antes utilizada, o hash da senha era disponibilizado no export do TC

Isso permitia backup/migração facilitada

Mas também permitia possível quebra dos hashes, que usam algoritmo obsoleto

Agora o hash é omitido, e uma nova estratégia de contingência é necessária; por enquanto se usa um export especial com o hash, mas o objetivo é fazer isso via PSQL

Impacto #6 do IRRd 4.2: Object class-filter

No IRRd 4.2, é possível tanto estabelecer filtro de classes de objeto quanto múltiplas importações para um mesmo source

Isso foi usado para, nos IRRs que publicam classes separadamente, espelhar apenas as classes que fazem sentido em IRR

Para os que publicam arquivo monolítico, o object class-filter foi usado para importar apenas esses objetos, a saber:

['as-set', 'aut-num', 'filter-set', 'inet-rtr', 'key-cert', 'mntner', 'peering-set', 'route', 'route6', 'route-set', 'rtr-set']

Mudança 2021 #4: Python nos scripts próprios

Inserção de maintainers e outras tarefas do TC eram automatizadas com bash shell scripts

Todos os scripts foram reescritos em Python (CPython) e novos scripts passaram a ser feitos também em Python

Mudança 2021 #5: Novo script de inserção

Consulta RDAP no Registro.br ao invés de WHOIS

Possibilidade de escolher qual dos contatos do ASN cadastrar no IRR

Importação de anúncios de roteamento (via RIPE RIS)

Criação automática de AS-SETs para anúncios, clientes e upstreams, parcialmente preenchidos

Mudança 2021 #6: Nova página do maintainer

Foi criada uma página para os que já são mantainers no TC IRR, com as opções de redefinição do maintainer (senha e contatos) e de remoção do TC IRR

A redefinição de maintainer resolve tanto situações de perda de senha, quanto de requisitos atualizados do IRRd 4.2 sobre os contatos

A remoção automatizada facilita a saída de quem preferir outro IRR, mas só pode ser feita com ASNs ainda existentes no Registro.br. ASNs que sejam removidos no Registro.br acabam sendo removidos pelo processo de limpeza da base.

Mudança 2021 #7: Remoção rápida de objetos indevidos

O TC IRR sempre teve uma clara postura contrária a criação de objetos proxy (objetos criados por um mantenedor que não o AS, rotas com origem diferente da alocação)

Essa limpeza era feita manualmente através de shell scripts

Ainda há um script manual de limpeza com checagens mais detalhadas, e que retira por exemplo os objetos que foram removidos no Registro.br

Mas agora há também um script online de checagem near-real-time (“MirrorKill”) para todo objeto adicionado ou modificado, que reconhece objetos proxy e os remove poucos segundos após sua inserção

Mudança 2021 #8: Status page

Monitoração de serviços foi adicionada no Uptime Robot

Isso permitiu também a criação de uma status page, http://status.bgp.net.br

Poderia ser melhor se o uptime da solução de monitoração fosse melhor

Mudança 2021 #9: Retorno do source REGISTROBR

O TC já teve um pseudo-source que continha as alocações do Registro.br, que então era o source NICBR

IRR da NTT implementou isso também, com nome REGISTROBR

Geração automática desse source foi reescrita em Python e com nome REGISTROBR para padronização entre IRRs

Mudança 2021 #10: Menor tempo de resposta a e-mail

Mesmo ainda sendo best-effort, o tempo de resposta a e-mails para db-admin@bgp.net.br foi bastante melhorado

Regras de procmail tem ajudado a separar mensagens automáticas

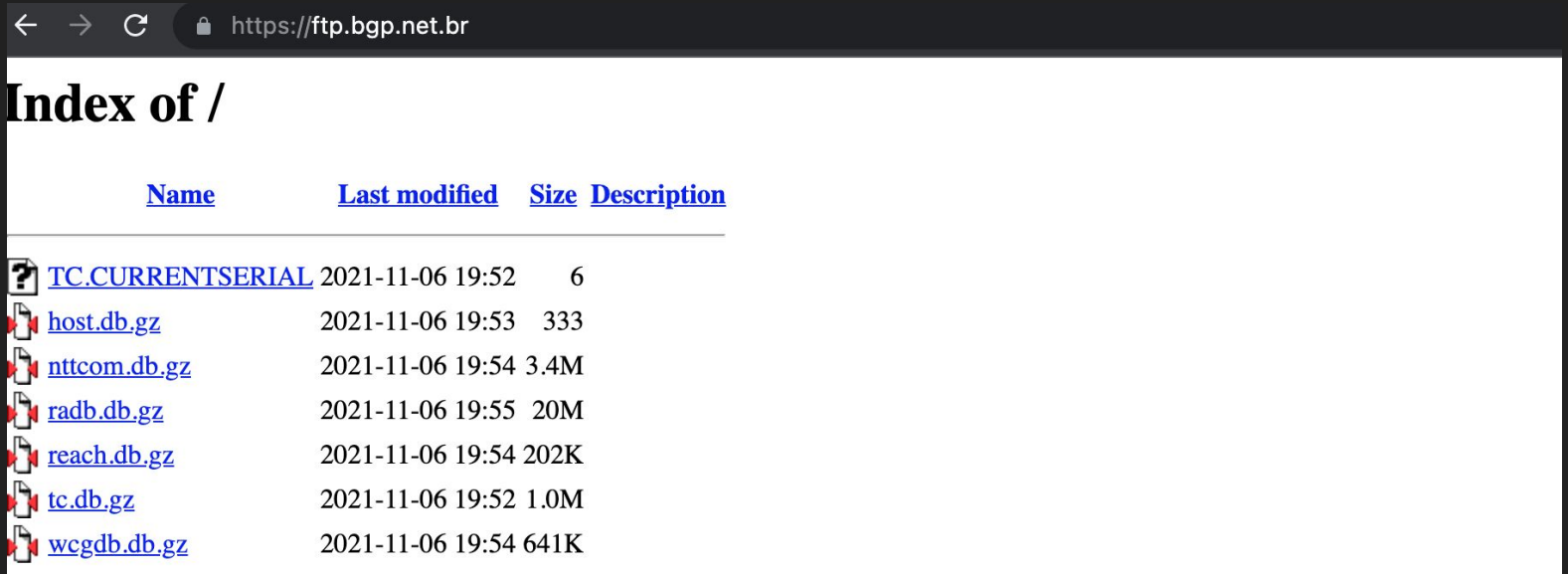
Ainda há um número significativo de e-mails com objetos RPSL (que deveriam ir para auto-dbm@)

Esperamos mudar alguns detalhes dos e-mails de cadastro para melhor orientar usuários

Mudança 2021 #11: Servidor de arquivos via HTTPS

Para uso não autenticado o protocolo FTP não é um problema de segurança, mas não agrega garantia de identidade

Arquivos agora podem ser acessados via HTTPS



Planos

Adicionar consulta ao PeeringDB para cadastro/verificação/análise

Replicação de PostgreSQL para operação multi-master do TC IRR

Edição Web de objetos

Retorno do analysis e do report

Verificação de congruência de objetos com outros IRR, quando houverem

Upgrade para 4.3, quando estável

That’s all, folks!

Dúvidas ? (Ou agora, ou por e-mail para db-admin@bgp.net.br)

Doações: https://www.buymeacoffee.com/faleiros (total recebido até agora: R$0)