RXGK

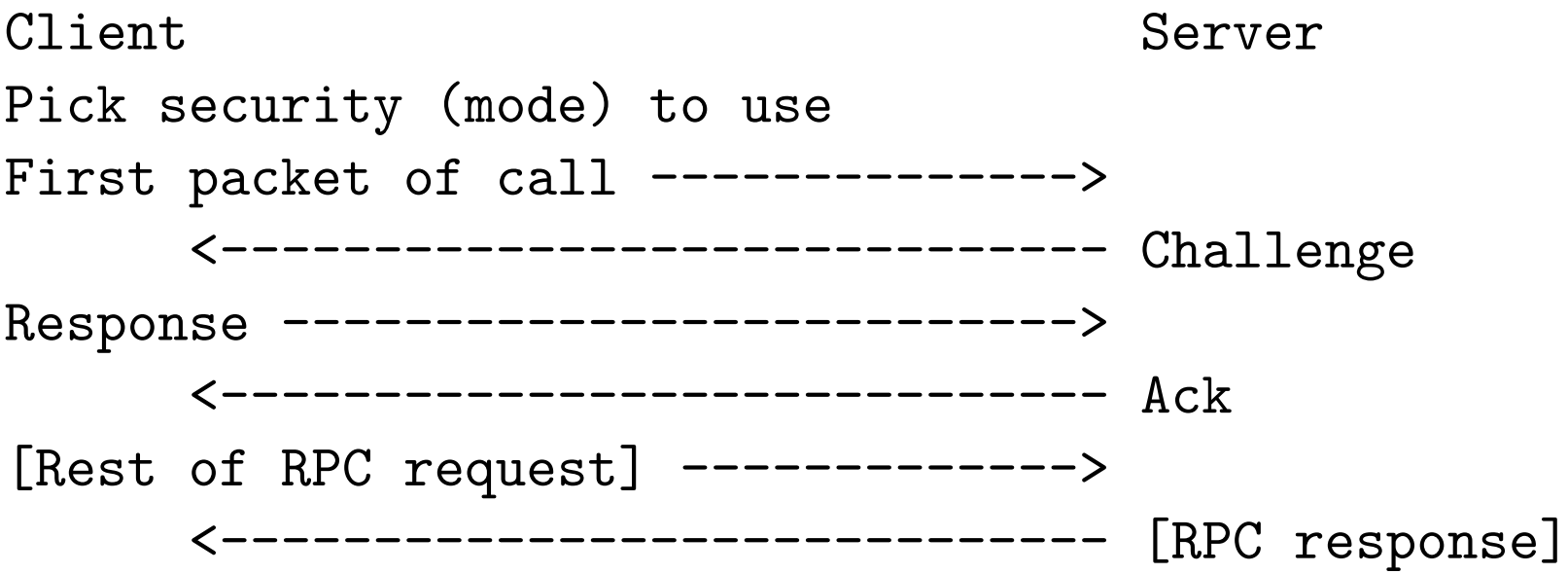
Benjamin Kaduk

20 June 2019

rxkad is bad

- roots in Kerberos 4, modified piecemeal as the ecosystem moved around us
- 56-bit keys, susceptible to brute force for O($100).
- confidentiailty protection for data transfer is painfully slow
- integrity-only is basically the same CPU cost as confidentiality
- server can't tell client to use encryption (or not)

Rx security refresher



rxgk is good

- Uses the same crypto primitives as Kerberos 5 (extensible!)
- Including 128- and 256-bit (AES) keys
- AES is faster than DES (and hardware acceleration is common)
- integrity-only HMAC is lower overhead than AES+HMAC
- separate negotiation (RPC) for crypto parameters

What is rxgk?

Over the (many) years, "rxgk" has come to encompass several things:

- encryption/MIC of data on the wire with krb5-level of protection
- robust key hierarchy
- GSS-API for initial authentication
- explicit negotiation of security parameters/capabilities
- combined host/user credentials
  - prevents cache poisoning
  - allows for richer ACLs
- per-fileserver keys
- ...

What is rxgk?

...

- full support for multi-instance Kerberos principals
- non-Kerberos GSS mechanisms
- secure callbacks
- flexible/extensible token format
- opportunistic security for anonymous clients

Specific mechanisms

Concretely, this means rxgk provides:

- packet-level protection routines
- new RPCs for negotiation/obtaining tokens
- new RPC for registering per-fileserver keys (which needs a new authorization model as well)
- a solution on top of existing key agreement techniques like anonymous PKINIT

rxgk-phase1

The “rxgk-phase1” topic provides:

- the core packet-protection routines
- printed tokens from the cell-wide key
- strong protection for localauth and (most) server-to-server comms

What's in 1.9.0?

The 1.9.x release series would be "development releases", a (relatively) rapid release cycle that is essentially snapshots of `master`. So, no prereleases; some releases may be very buggy/unstable; bugfixes are just in the next snapshot.

- rxgk-phase1
- options for vlserver/ptserver to use rxgk to each other
- (but not one to *require* rxgk between each other)
- options to use rxgk for vos and pts (localauth) queries to the dbservers
- `asetkey -random` to generate rxgk keys

What's not in 1.9.0?

We've got a plan and some rough code for:

- user authentication to dbservers
- ways to use security parameter negotiation
- authentication to fileservers (and thus, use of host credentials)

We still need to finish designs for:

- per-fileserver keys (new vldb format)
- secure callbacks

How can you help?

- Please run 1.9.x on your test cell
- Add to the test suite
- Document procedures to roll out rxgk (phase 1, phase 2, . . .)
- Document (hypothetical) procedures to operationalize a vldb format change in your environment/workflow
- Code review
- pthread conversions
- other code maintenance (convert utilities to positional arguments, type consistency for vnodes/IP addresses/etc., fileserver autotuning, . . .)

Questions

Thank you!