GnuGk – The GNU Gatekeeper

OpenSource Telephony Summit 2005

Jan Willamowius

Michal Zygmuntowicz

Agenda

- The GNU Gatekeeper project
- Feature overview
- Deployment Scenarios
- Configuration
- Performance
- The Future
- A Wishlist

How did it all start ?

- started in 1999
- version 1.0 released in 2001
  - usable for small production setups
- version 2.0 released in 2002
  - it is being used in many larger installations
- version 2.2 released in October 2004
  - architectural redesign, more scalable
  - new features like routing policies etc.

GnuGk Community

- 2 mailing lists
  - 900 members on the users mailing list
  - 350 members on the developers mailing list
- 8000 downloads / month
- 25000 visits at **gnugk.org** / month

Feature Overview (1)

- GPL licence
- support for Unix (Linux, Solaris, FreeBSD), MacOS X and Windows
- can be run as a Windows service
- H.323 protocol handling through OpenH323 library
- all regular H.323 gatekeeper features
  - address translation (alias to IP)
  - access control, call authorization, accounting
  - call routing
  - etc.

Feature Overview (2)

- NAT traversal - both outgoing and incoming calls
- authorization and accounting with various backend systems (plain text file, SQL, LDAP, Radius)
- telnet monitor and admin interface
- gatekeeper clustering and failover support
  - child-parent gatekeepers
  - neighbor gatekeepers (interzone communication)
  - alternate gatekeepers

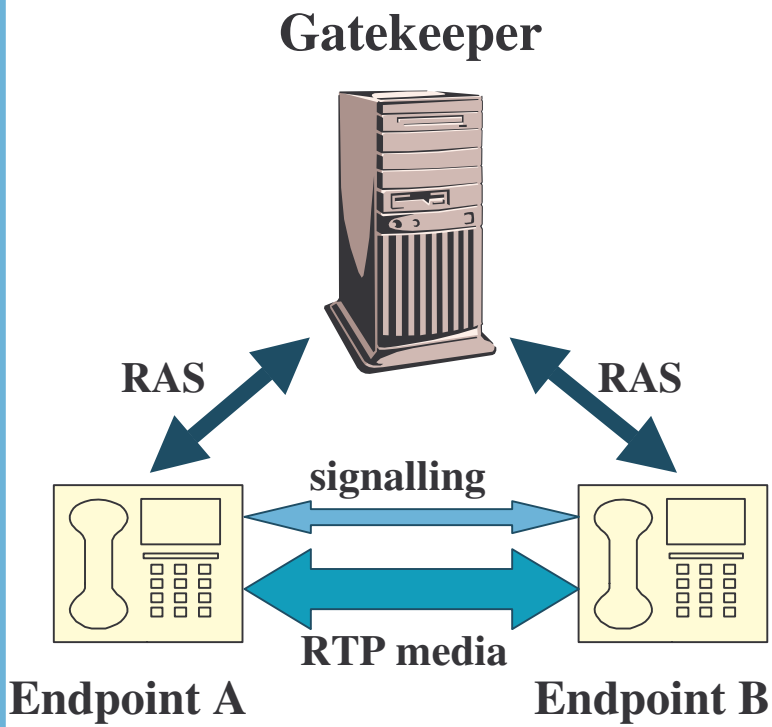
Feature Overview (3)

- support for various versions of H.323 protocol (V1 endpoints, some V4 features)
- H.235 security (authentication) – try GnuGk 2.0.9 !
- CTI functions:
  - inbound call routing (“virtual queues”)
  - call transfer
- E.164 number rewritting (call routing)
- configuration changeable at runtime
- Graphical User Interface in Java for monitoring

Operational Modes

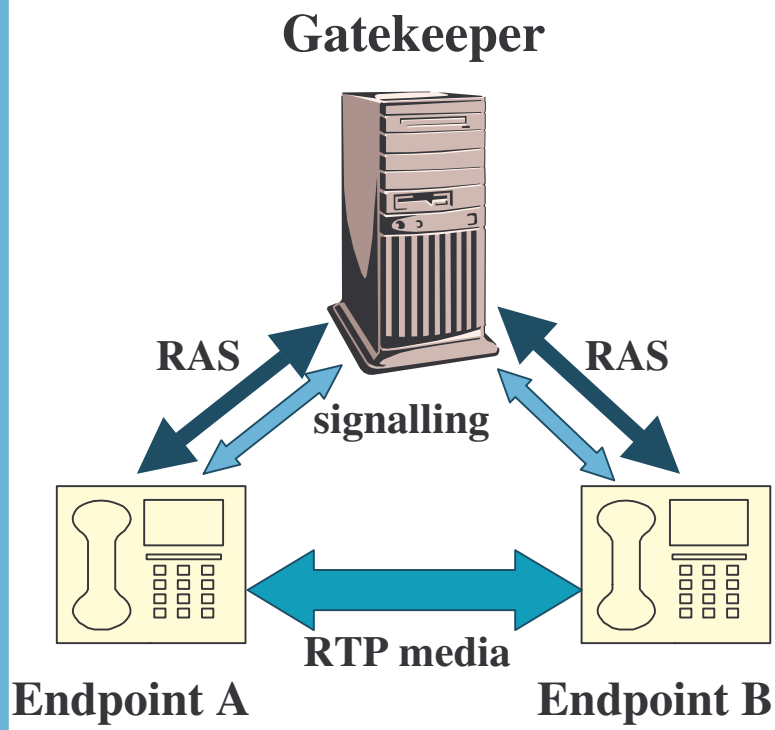
- direct signalling mode
- gatekeeper routed signalling mode
- full proxy mode (signalling + RTP media)

Direct Signalling Mode



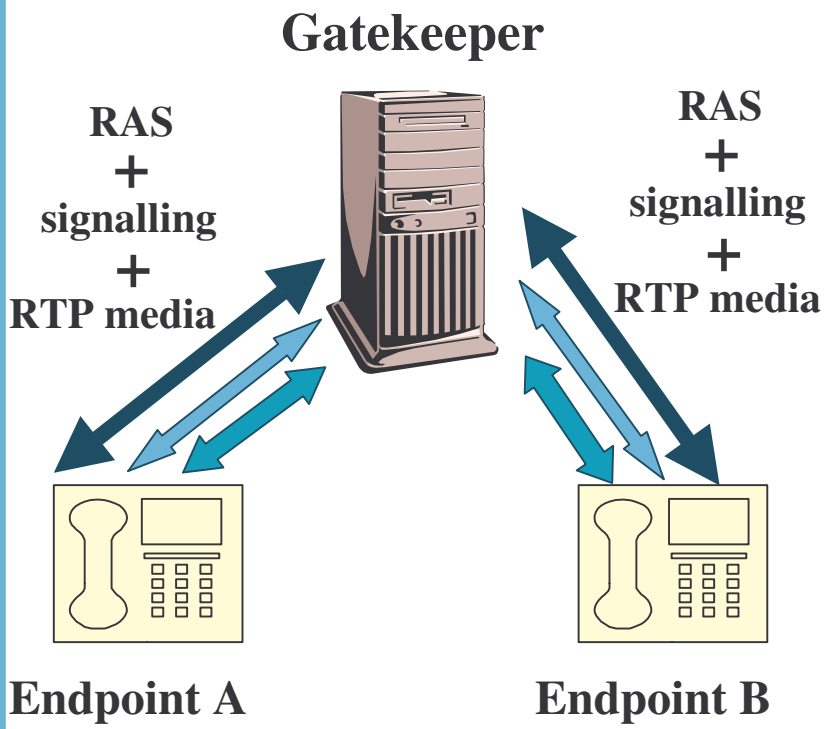
- only RAS channel between endpoints and the gatekeeper
- signalling directly between endpoints
- very good scalability
- lack of precise call control

Gatekeeper Routed Signalling



- signalling channel is routed though the gatekeeper
- precise call control (authorization, accounting)
- additional services like call transfer
- good balance between peformance and flexibility

Full Proxy Mode

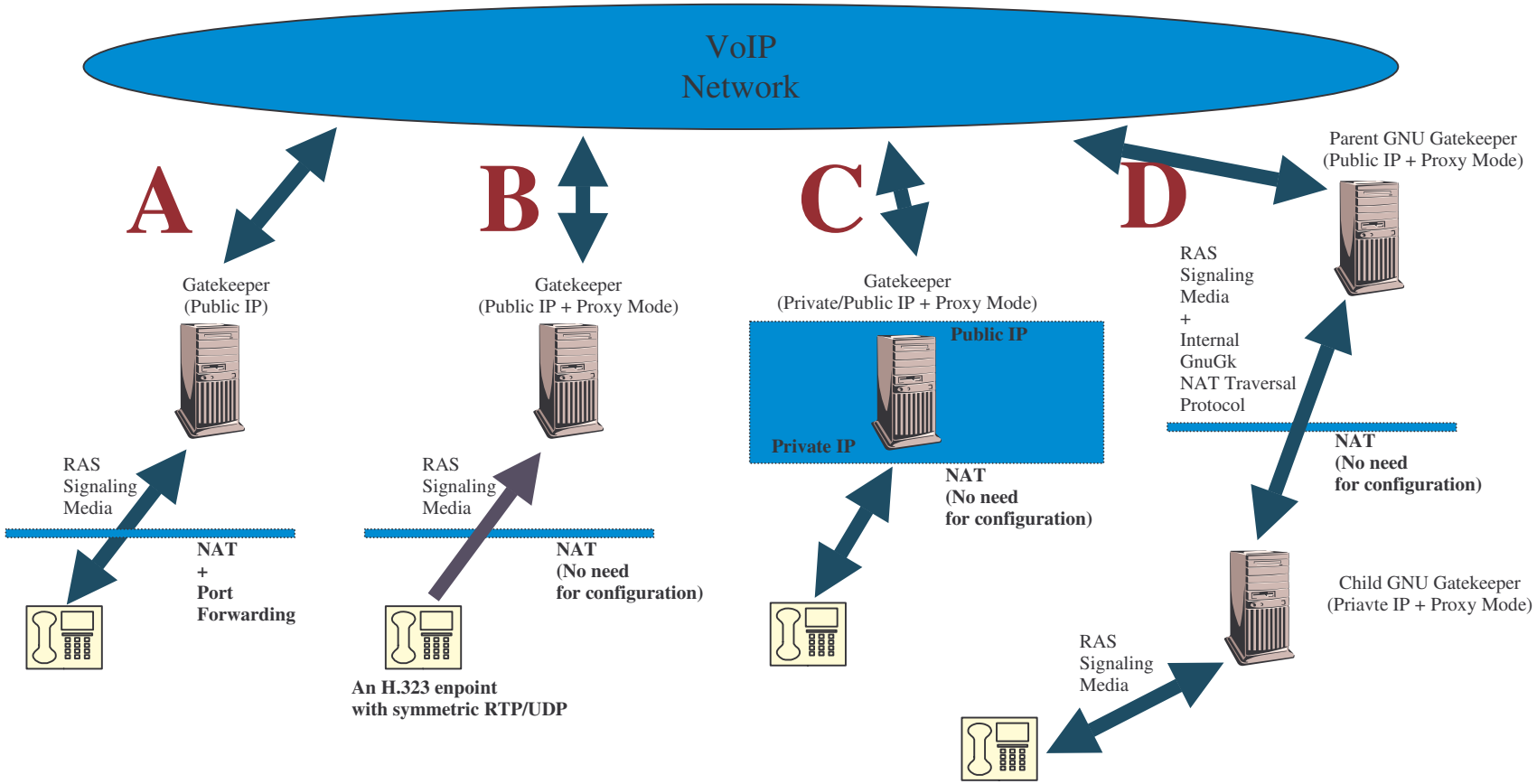


- all data (RTP audio, RTP video, T.120 data) is routed though the gatekeeper
- no direct communication between endpoints
- high CPU/bandwidth consumption
- designed to allow firewall/NAT traversal

NAT Traversal (1)

- 5 possible scenarios:
  - endpoint behind NAT, port forwarding enabled
  - (outbound calls only) endpoint behind NAT that knows how to use symmetric RTP UDP, gatekeeper in proxy mode
  - gatekeeper (proxy mode) on a NAT box (with access to both internal and external network interfaces
  - gatekeeper behind a NAT box, registered as a child with a parent GNU Gatekeeper (both have proxy mode enabled), both use internal NAT traversal protocol
  - endpoint behind NAT, knows how to use symmetric RTP/UDP and internal GnuGk NAT traversal protocol

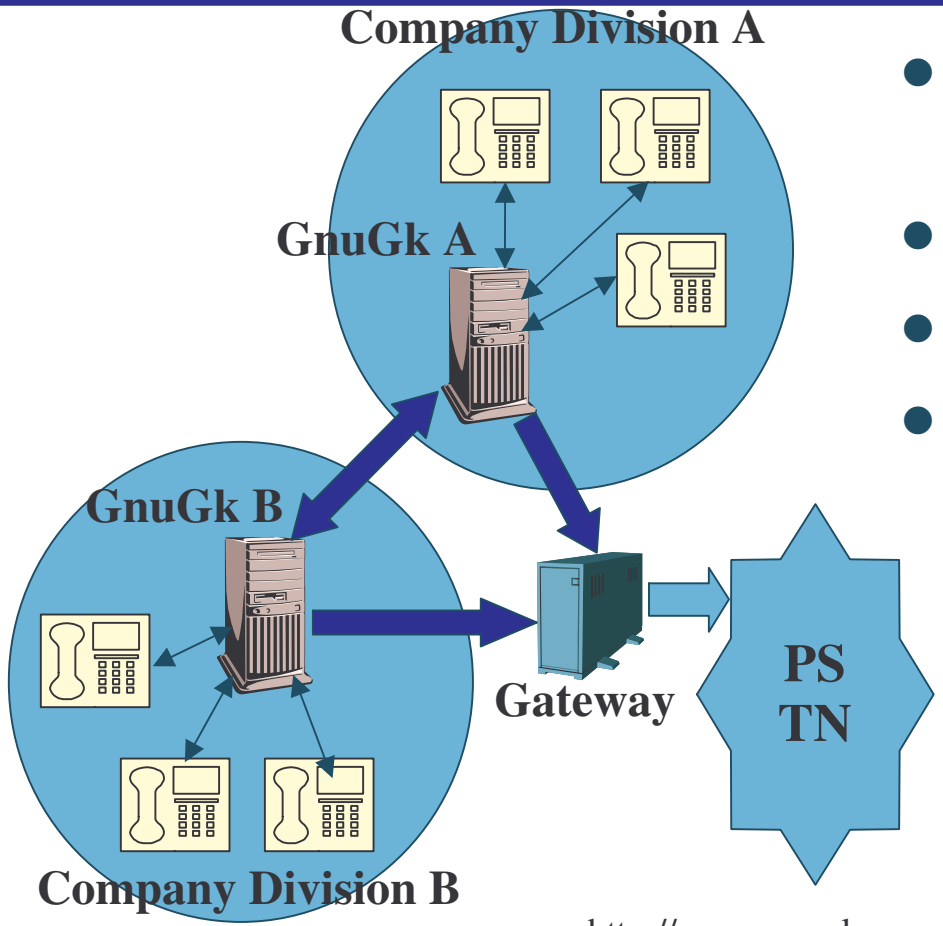
NAT Traversal (2)



GnuGk Deployment Scenarios

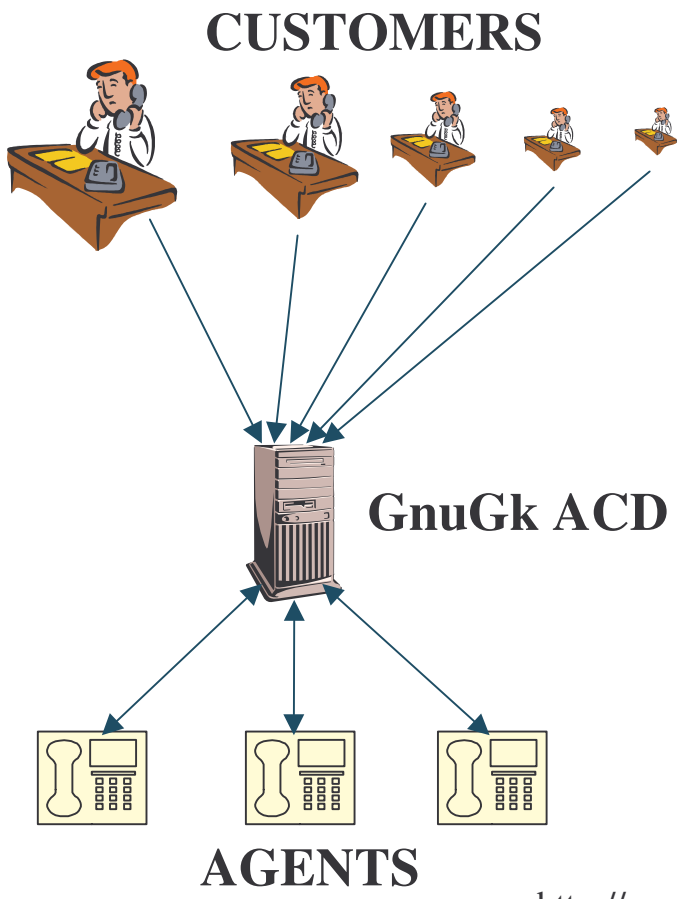
- PBX Replacement
- Prepaid VoIP Telephony
- Call Center
- Call Termination Services
- and much more ...

PBX Replacement



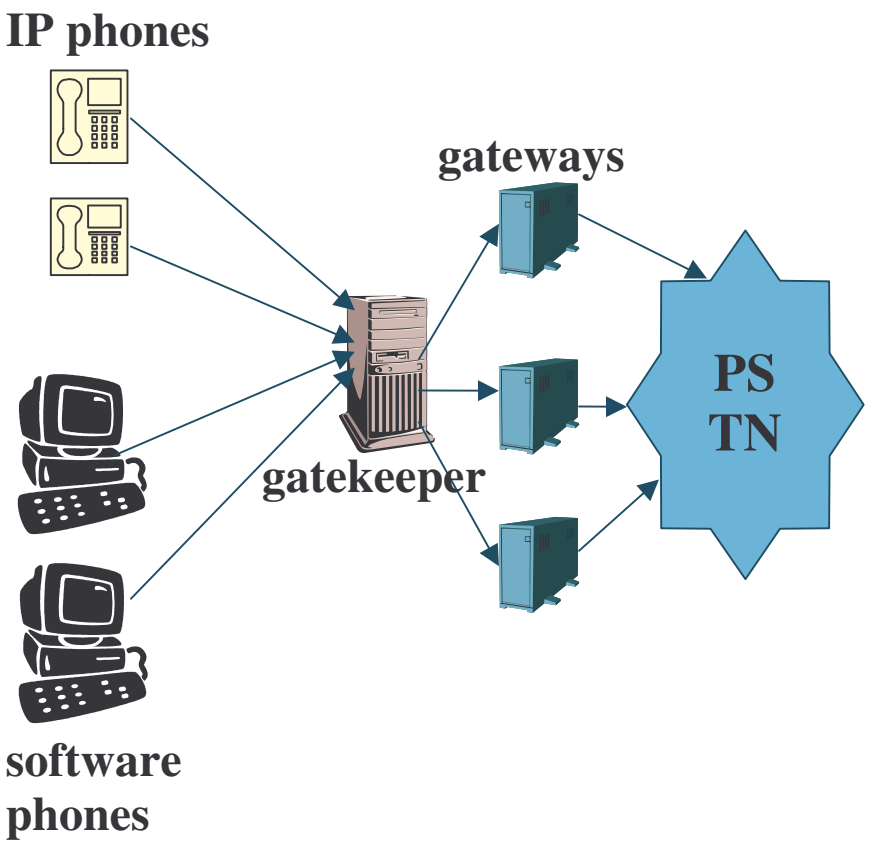
- internal calls within the company
- inter-division calls
- numbering plans
- cheap PSTN calls

Call Center



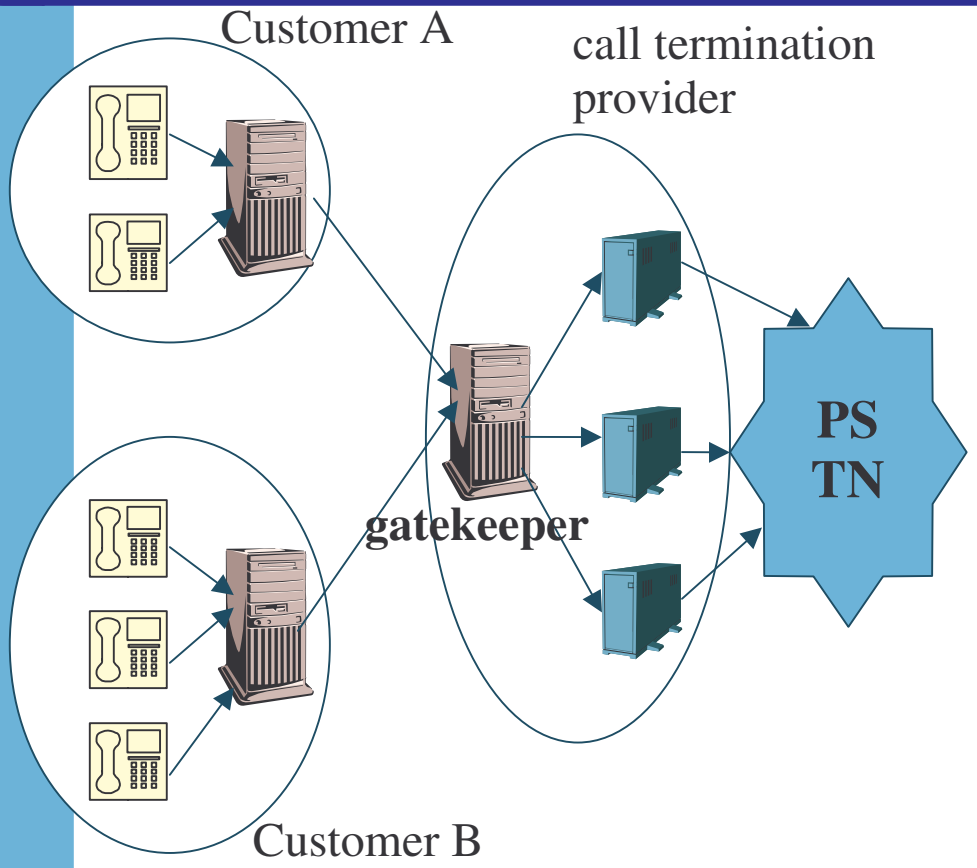
- ACD application (Automatic Call Distribution)
- calls to a single number are distributed to many agents (eg. hotline)
- various call distribution policies:
  - longest idle
  - first idle
  - round robin
  - TODO: skill based

Prepaid Calling



- call authorization and accounting
- enforcing limit on call duration
- easy integration with Radius and existing billing systems
- can be easily built from open source components only:
  - GnuGk + Radius server + SQL database

Call Termination Services



- call authorization and accounting (gatekeeper routed signalling mode)
- call routing decisions:
  - route the call to a specific gateways
  - route the call other call termination providers

GnuGk Configuration (1)

- manual in download archive (chapter 3 is a short tutorial)
- all configuration settings are read from a text file
  -
```
[Gatekeeper::Main]
Fourtytwo=42
Name=GnuGk

[RoutedMode]
GKRouted=1

[GkStatus::Auth]
rule=allow
```
- can reload changed config at runtime

GnuGk Configuration (2)

- the configuration is divided into sections:
  - global parameters
  - gatekeeper mode (direct signalling, routed signalling, full proxy)
  - neighbor/parent gatekeepers
  - routing (E.164 rewrite, gateway prefixes)
  - authentication modules
  - accounting modules
  - virtual queues

Accounting / Billing

- many acct modules
  - flat file (FileAcct)
  - Radius (RadAcct)
  - SQL (SQLAcct: PostgresSQL, MySQL)
  - Telnet interface (limited)
- SQL billing application for PostgresSQL in contrib/ directory
  - started as an example for an OSTS 2004 tutorial, now part of the GnuGk package
  - flexible billing **engine**

SqlBill (1)

- small but complete core for a billing/tariffing engine
- SqlBill provides
  - endpoint authentication by means of username/password, username/IP or IP only and alias control
  - endpoint/call authorization (allowed destinations, maximum call duration limit, account balance)
  - real-time account/call billing
  - support for prepaid/postpaid, originating/terminating account types
  - flexible tariffing engine

SqlBill (2)

- SqlBill does not provide
  - bussiness logic (invoicing, detailed customer data, payment processing, etc.)
  - user interface (minimalists can use pgAdmin;)
- technical details
  - can work on large databases
  - processes 50 calls / second on an average PC machine
  - communicates through RADIUS or directly with GnuGk
  - interfaces with PHP/.NET/ODBC applications easily
  - extendable to interoperate with other protocols/software

GnuGk Telnet Interface (1)

- the „status port“
- interface for humans and external applications
- interface to non-GPL code
- remote administration
  - configuration changes/reloads
  - gatekkeper statistics (endpoints, total / active calls etc.)
  - manual call disconnect and endpoint unregistration
  - username/password based access authentication
  - call routing (“virtual queues”)
- live CDR output

GnuGk Telnet Interface (2)

```
GkStatus: Version(1.0) Ext()
Toolkit: Version(1.0) Ext(basic)
Startup: Fri, 08 Oct 2004 00:59:03 +0100 Running: 80 days 01:34:23
;
s
-- Endpoint Statistics --
Total Endpoints: 230 Terminals: 157 Gateways: 73 NATed: 147
Cached Endpoints: 1 Terminals: 0 Gateways: 1
-- Call Statistics --
Current Calls: 54 Active: 49 From Neighbor: 12 From Parent: 0
Total Calls: 1946364 Successful: 764238 From Neighbor: 533765 From Parent: 0
Startup: Fri, 08 Oct 2004 00:59:03 +0100 Running: 80 days 01:34:23
```

Telnet Interface Applications

- Monitoring
  - Java GUI
  - GnuGk PHPStatus
  - OpenIP PBX (outdated)
- Call Routing
  - GnuGk ACD
  - custom routing / LCR applications
- Billing
  - interface to other billing applications
    - use other acct module if you can

Ways to route calls

- Gateway selection (config)
- Destination rewriting (config)
- Virtual queues (external)
- Radius based (external)
- use the new routing policies in 2.2 to configure which of the above are active

Virtual Queues

- no queued calls with announcements etc.
- „external ARQ rewriting“
- Config
  - define list or regexp of destinations to route
- Event
  - RouteRequest
- Commands
  - RouteReject (disconnect call)
  - RouteToAlias (change destination alias)
  - RouteToGateway (change destination alias and destination IP „out-of-zone routing“)

GnuGk Performance

- depends strongly on the gatekeeper mode selected (direct, routed signalling, full proxy)
- few performance statistics/tests (testers wanted!)
- direct and routed modes are able to process a few thousands of simultaneous calls on a typical high-end PC machine
- full proxy mode is designed for small call volumes - a few hundreds of simultaneous calls
- for large volume of calls the Unix version of GnuGk is recommended

Performance Optimization (1)

- use LARGE_FDSET=... for large call volumes
  - compiletime config
  - stresses CPU less than OpenH323 socket handling
  - LARGE_FDSET=1024 for <= 100 concurrent calls
  - rule of thumb:
    - max. concurrent calls * 10 + 20%
    - 10 sockets/call: 2 for Q.931 + 2 for H.245 + 6 for RTP etc.
  - usually an OS limit for maximum number of file handles per process needs to be increased (using 'ulimit' command, for example) to match the new LARGE_FDSET value

Performance Optimization (2)

- GnuGk spawns one or more threads (signaling handlers) to handle signaling messages and perform authorization / accounting
- for best call throughput (and max. concurrent calls) tune CallSignalHandlerNumber / RtpHandlerNumber variable
  - runtime config
  - Windows PWLib has default limit of 64 sockets / thread
    - or recompile PWLib with FD_SETSIZE=x macro
  - don't let a single signaling handler to handle too many calls
    - CallSignalHandlerNumber=ConcurrentCalls/10

Performance Bottlenecks

- slow accounting/authorization backend (a database without indexes or with inefficient ones, queries not optimized, no RAID disks, RADIUS server runs out of resources, etc...)
- excessive network packet throughput:
  - a single G.723.1 call requires (at most) 70 UDP packets/s to be sent/received (in both directions) from each party:
    - 140 packets/s per call => 45.000 packets/s for 300 concurrent calls
    - add 5% for signaling => ca. 50.000 packet/s for 300 calls
  - Gigabit Ethernet cards can handle high packet rates without triggering too much interrupts to the kernel

GnuGk Future (1)

- 2.0 branch
  - stable, well tested product
  - only bugfixes will be added
  - LDAP, H.235, MacOS X
- 2.2 branch
  - current stable version (2.2.0 was released Oct. 2004)
  - config is mainly backward compatible (except for routing policies)
  - redesigned to give much better performance and call routing control

GnuGk Future (2)

- flexible call routing:
  - failover support (multiple destination routes)
  - smart route selection (LCR – Least Cost Routing)
  - multistage E.164 number rewritting
- more advanced gatekeeper clustering
- dialler applications
- development of external applications on top of GnuGk

GnuGk Future (3)

- internationalization of the documentation
  - currently: English, Portuguese
  - comming: French, maybe Spanish
  - slighly outdated: Chinese (new maintainer ?)
- TAPI, JTAPI support

A Wishlist

- please link to **gnugk.org** so others can find GnuGk
- please send us your
  - feedback
  - config tips for endpoints, gateways etc.
  - success stories
  - tools (big or small)
  - whatever else you have to share with the community

Visit http://www.gnugk.org

Thank you!