Securing syslog on FreeBSD

© Albert Mietus
albert@ons-huis.net
albert.mietus@pts.nl

Agenda

- Syslog
  - de-facto standard, characteristics
  - old, insecure
- Crypto (short)
  - asymmetric keys (public // private keys)
    - SHA & DSA
- Secure syslog on FreeBSD
  - Correct (unchanged) forwarding and storing
  - a (simple) syslog-sign implementation

© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

ALbert Mietus

That's me ...

- Employed by PTS Software, NL
  - Technical Software & Infra. Eng.
    RT/Embedded, Telecom, ...
    Security, Unix, …
  - Consultant

- *This code/presentation is "hobby"*
  I'm allowed to give the code away

© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

Part 1

syslog, an overview

© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

Syslog (1)

Question: Who writes the /var/log/<files> ?

AND WHY those FILES ??

They are written by syslog(d), which is

- configured in /etc/syslog.conf
  - "routing table" for log-messages to files, users, systems. ...
- assisted by `newsyslog` (on BSD)
  - moves old messages to .[0-9].gz and expires them.

© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

Syslog (2)

- Syslog is the de-facto standard of logging
- It consists of a daemon, a API (in libC) and a protocol: RFC3164 (last year)
- Syslog is: old, widely used and flexible
- Syslog(d) uses:
  an UDP protocol
  & plain text-files.

© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

Syslog, Again ..

Question: Who writes the /var/log/<files> ?

ARE YOU SURE ???

- Just maybe, somebody edited the file ..
- Just maybe, somebody sent a fake log
- Just maybe, it is all correct.

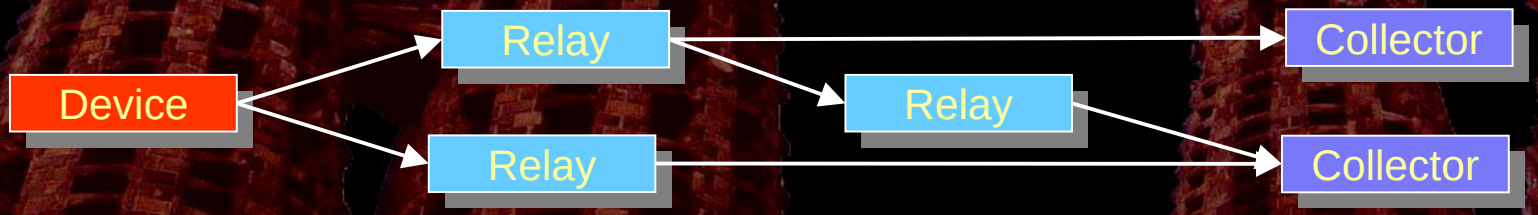
But can you assure it?

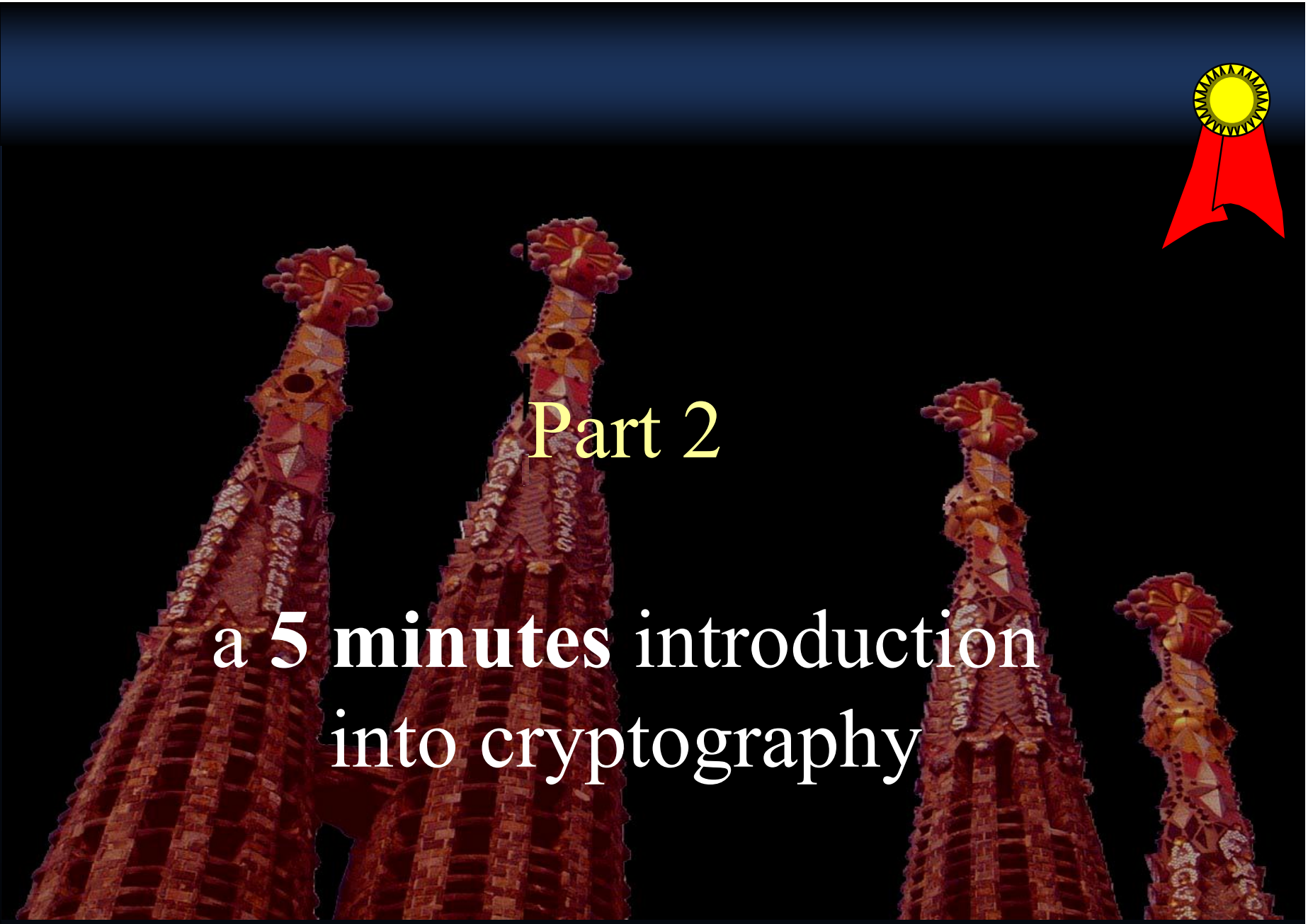
Syslog-secure

- Syslog is fine, but we need to secure it
- There is a IETF working on this:
  - rfc3164: describes the average “as-is” protocol
  - rfc3195: uses a secure protocol: BEEP (-reliable)
  - syslog-sign: insert digital signatures (draft-07)

– syslog-sign

Increases security of the /var/log/<files> too!





Part 2

a **5 minutes** introduction into cryptography

© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

Cryptography

- *Crypto* is used for two purposes:
  - to hide information
    - » Examples: https, ssh, passwords
  - for authentication : hashes/signatures
    - » Examples: MD5-hashes, PGP-signatures, ..
- Two kinds of crypto-algorithms exist:

Symmetrical	e.g.	DES (passwd)
Asymmetrical	e.g.	RSA

I explain asymmetrical authentication only

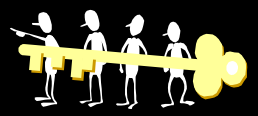
© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

Asymmetric key authentication

Encrypting is done with a private key

Decrypting is done with a public key

NB
It's impossible to calculate the "other" key !!!



Some text you can read	Private key	Fbzr grkg lbh pna ernq	Public key	Some text you can read

Result: you are sure who SENT it ! ! !

Hashes & Signatures

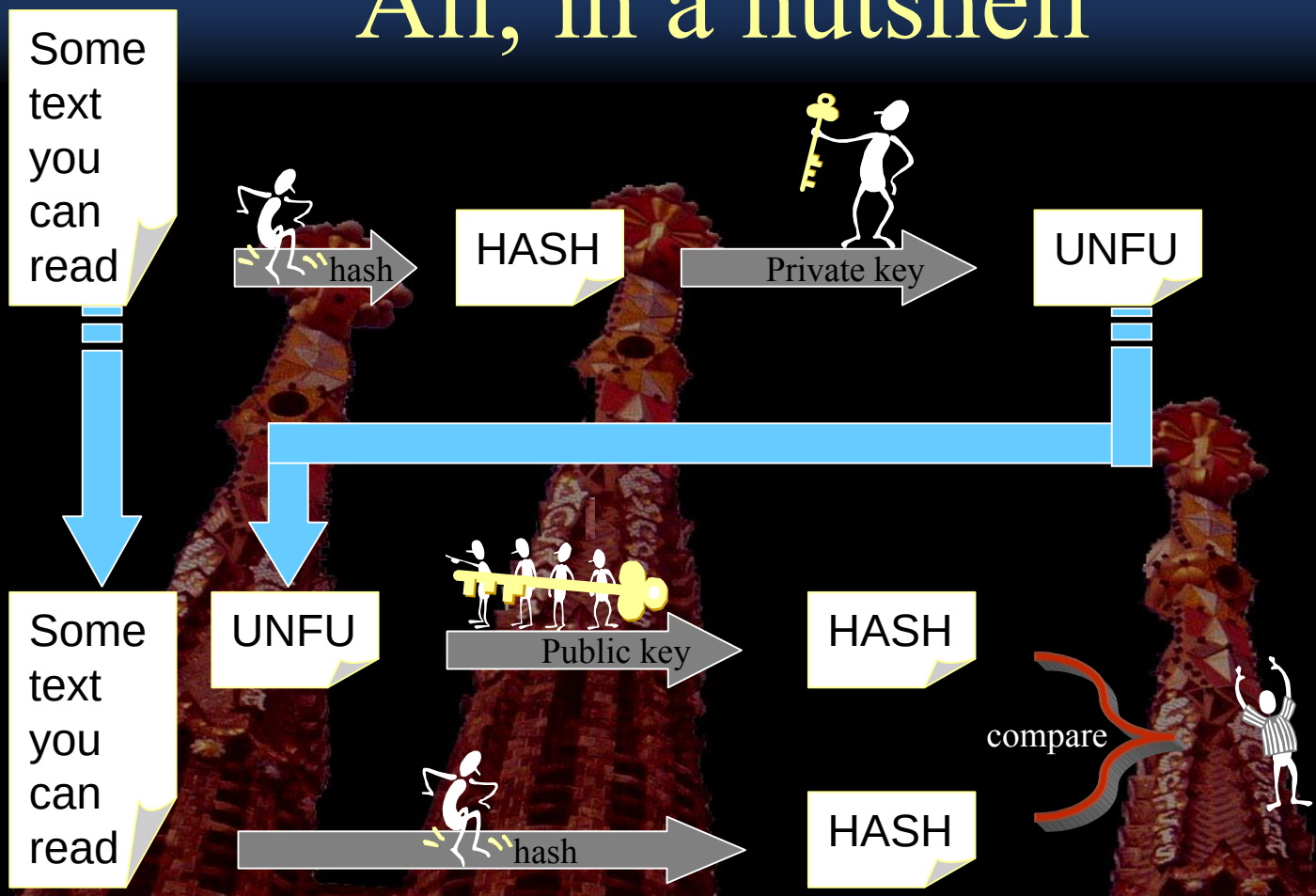
Hash: (of text)

- A kind of cryptographic checksum
- It's impossible to change "*text*", without its *hash*!
- A hash has a **short**, **fixed** length
  - » Example: SHA1-hashes are always 160 bit

Signature:

- Instead of crypting "*text*" directly, use its "*hash*"
  - – It is shorter, so faster
- Always done with asymmetric keys
  - » As shown before

All, in a nutshell



IFF the HASHes are equal, the message is ***OK*** ! ! !

© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

Part 3

securing syslog,
using syslog-sign.
A simple implementation

© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

Step by step

- Syslog-sign is an extension on “rfc3164”
  - First we need a RFC3164-compliant syslogd
    - » This is shown in a handful of sheets
- Then,
  - We need to insert signatures
    - » Again a few sheets
  - We need to distribute the public key
    - » A quick hack, in 1 sheet
  - We need to verify the signatures
    - » (to be done)

© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

Step 1: "rfc3164 mode"

Syslog-sign adds *detached*, *in-band* signatures

Requirement: Never change a message !!

Design: Use RFC3164 format,
all the time

rfc3164-mode: **design**

FreeBSD's forwarding changes the message

- New: Use rfc3164 format in sending
- Also: Flag received messages with 'rfc3164'
  - » It's a kind of "do not change" flag
  - » Check the message, and rewrite as rfc3164 says

The logfiles don't store the PRI, by default

- New: Option "-v -v -v" stores them *at the front*
  - » In '**<**' [digit] '**.**' [digit] '**>**' format

☞ The interface with "libC" isn't changed.

- » no header (only pri & msg) ==> not rfc3164 compliant

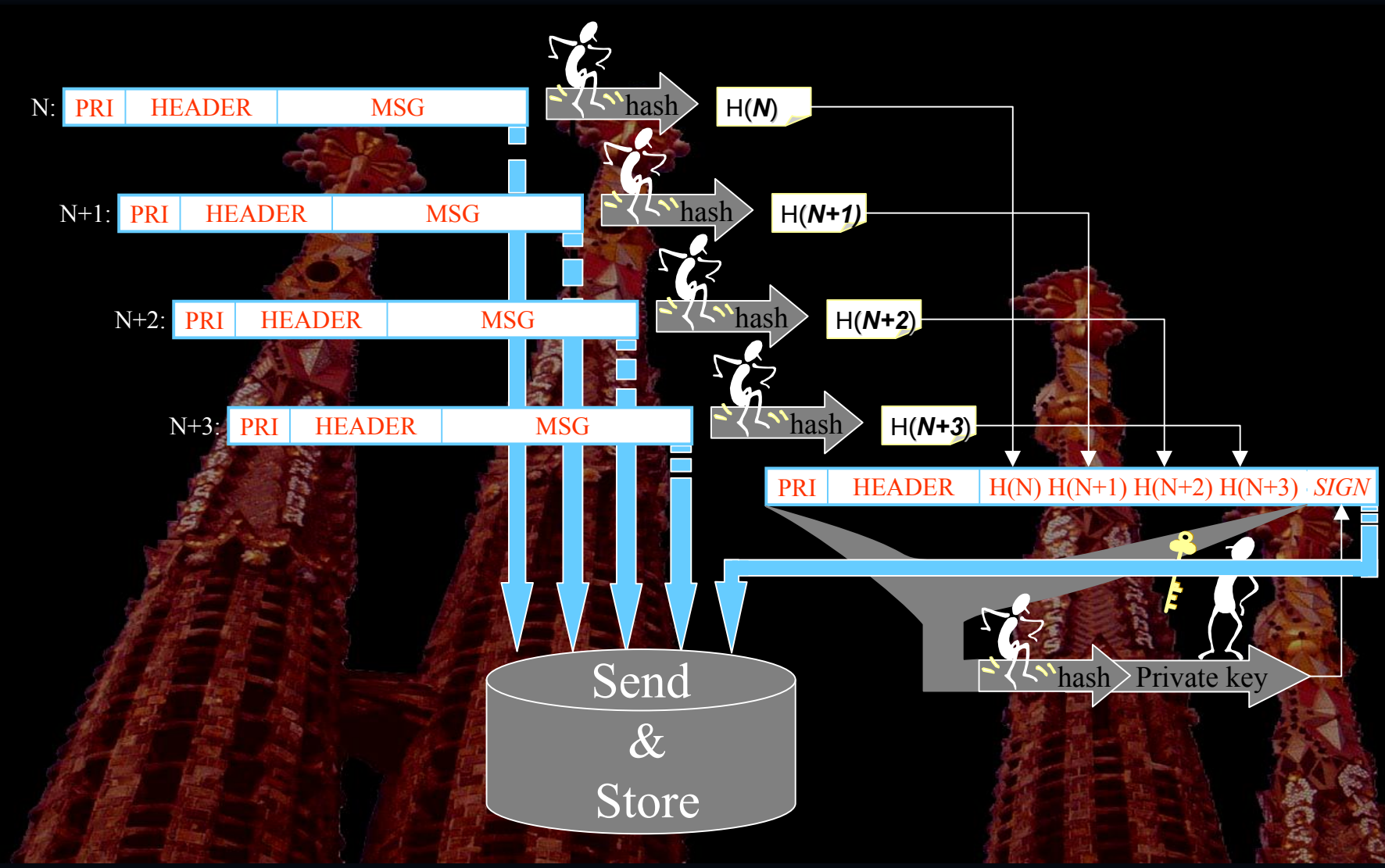
Rfc3164-mode: **implementation**

- One function is split into 2 parts
  logmsg() into logmsg() and dispatch()
- "Logic" is moved, to concentrate it
- Documentation is added
  - » Generally a good idea

☞ As little changes as possible

⇨ 100% compatible
with existing use of FreeBSD systems

Syslog-sign: **concept**



© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

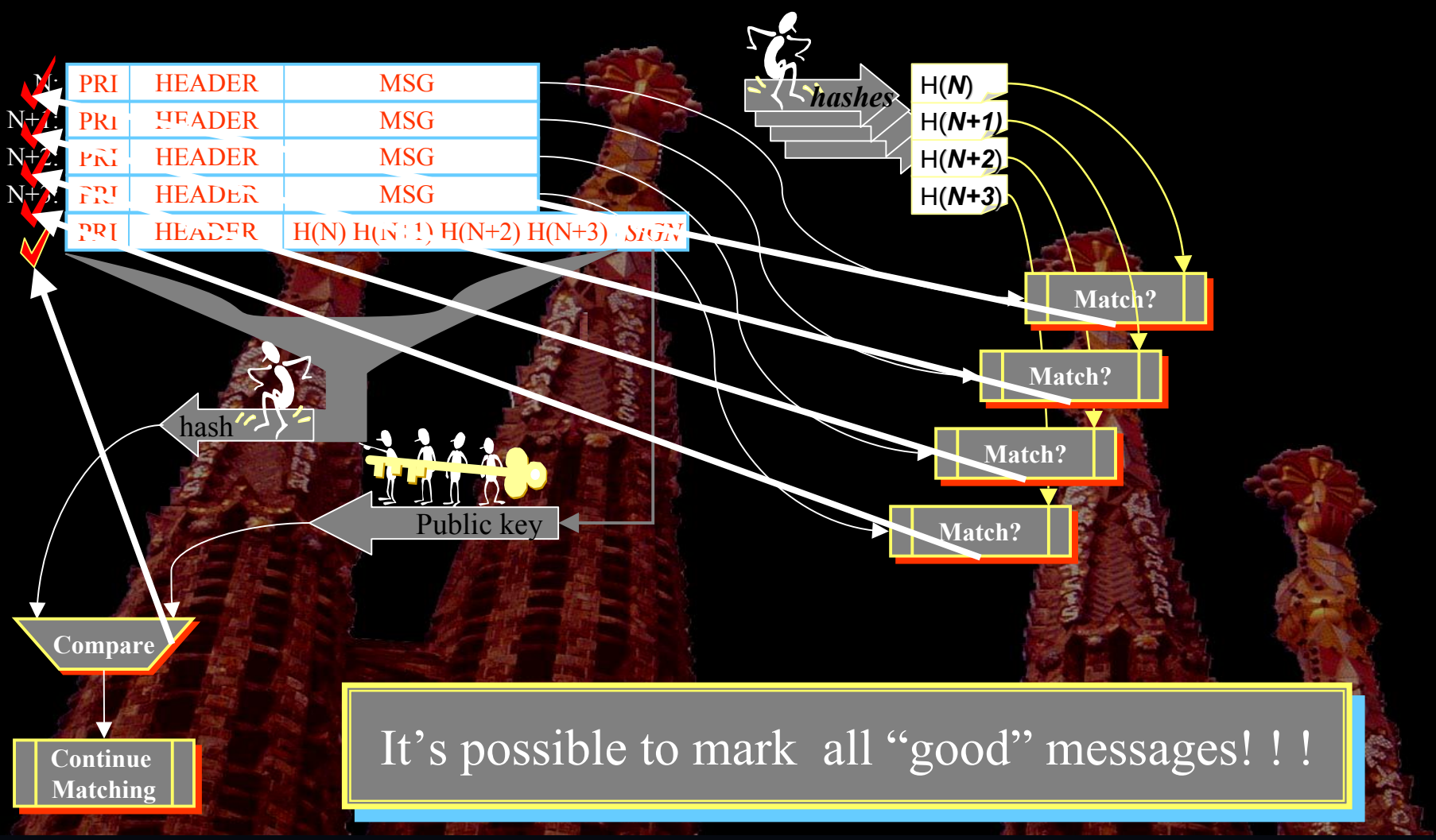
Syslog-sign: **transport & storage**

N:	PRI	HEADER	MSG
N+1:	PRI	HEADER	MSG
N+2:	PRI	HEADER	MSG
N+3:	PRI	HEADER	MSG
	PRI	HEADER	H(N) H(N+1) H(N+2) H(N+3) *SIGN*

All messages are handled as normal syslog messages.

They are transported and stored by all (rfc3164 compliant) syslog daemons !

Syslog-sign concept: **verifying**



Syslog-sign, details

Messages are sent over UDP, they can get lost

- But we should**n't** lose … (because, we)
  - Signature blocks: (can't check upto 25 messages)
  - Certificate blocks: (can't verify at all!!!)

Therefore, redundancy is used

- Certificate is send periodically
- Signatures are sent in multiple *"sliding"* blocks

Additional advantage:

Deleting a few (stored) messages can be seen.

© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

signed in syslogd

All *new* messages should get a signature

» We assume the forwarded ones have one already

- The signatures are calculated in syslog**d**
  - Doing it in "libC" is an option, but more complicated
    - » This would change the libC ↔ syslogd interface
- Syslog**d** does not verify messages
  - This should be done offline
    - » Then the *store* is verified also

Trivially: "rfc3164-mode" should be on.

Syslogd-sign, design

- At start-up, a (DSA) keypair is generated
  - The public key is sent with Certificate blocks
  - The private key is stored solely in memory
- Each new message is flagged 'sign'
  - » When read by /dev/log/ or /dev/klog or similar

On dispatching calc & store a hash (when "sign")

- » Don't change the message afterwards!
- » Store a "redundancy" down-counter with it

On awaking, send pending signature blocks

- » Another timeframe ==> probably we don't lose both

© ALbert@ons-huis.net
ALbert.Mietus@pts.nl

Syslogd-sign, implementation

- Need to move all logic out of fprintlog()
  dispatch(), now works on base of rfc-messages
  - » dispatch() is the 2nd part of logmsg()

  (So,) logmsg() needs to format the message.
  - » Now most logic is concentrated there!! It has become complicated

  – The header has to be stored in "struct filed" too

- Code split in 2 files: syslogd.c & sl_sign.c
  – The have a quite narrow interface

Complication: "compression"

- FreeBSD's syslog decreases the number of messages, by counting repeating messages
  "*Last message is repeated XXX times*"
  – This reduces network, storage & "*admin*" load

When signing, it becomes more complicated

– Compression is done per outlet
– Signing is done at the input

Result:

No 1:1 relation hash-blocks <==> messages

A solution: No compression

The public key

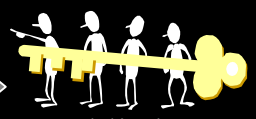
- The public key, base64-encoded, is sent fragmented in some syslog messages.
  Also: a timestamp & signature
- As a new one is sent each reboot, it's not 100% secure
  It is resent once in a while
- However,
  no administration is needed!

Base64 is used to send a binary part

PRI	HEADER	@#SigCer Base64
PRI	HEADER	@#SigCer is used
PRI	HEADER	@#SigCer to send
PRI	HEADER	@#SigCer a binary
PRI	HEADER	@#SigCer part

Public key & verification

PRI	HEADER	@#SigCer Base64
PRI	HEADER	@#SigCer is used
PRI	HEADER	@#SigCer to send
PRI	HEADER	@#SigCer a binary
PRI	HEADER	@#SigCer part *SIGN*



Public key

Validation can start after the public key is assembled from Certificate Blocks

PRI	HEADER	H(N) H(N+1) H(N+2) H(N+3) *SIGN*

The key is needed to validate the signatures

both of the cert and sign blocks

PRI	HEADER	MSG
PRI	HEADER	MSG
PRI	HEADER	MSG
PRI	HEADER	MSG

But they don't need to be "in order"!

Result

- Without admin, or additional configuration:
  - Certificates and signatures are generated
    The are send, transported & stored automatically
  - Offline verification becomes possible (at any time)

  Drawback: RFC3164 mode should be "on" (-V-V-V)
  This results in a slightly different file format.

- It's a drop-in replacement in most cases

You can assure the log-messages are correct ! ! !

And finally ...

- The Code, based on FreeBSD-4.*
  - an rfc3164-compliant version is available
  - an alfa-version of syslog-sign is available
    - » Some pasts to be implemented // debugged
    - » The (draft-07) rfc needs to be updated
  - More:
    - A TCP-transport extension needs to be integrated
    - A Kerberosed extension is under study *("delayed")*

- Mailto:albert.mietus@PTS.nl Subject: syslog

for a copy or for info (or support)