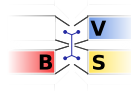


Syslog Protocols

Martin Schütte



22/29 May 2008

Introduction

- Logging

- Syslog vs. Audit

BSD Syslog

- Overview

- Format

- API

- Daemon

- IPC

IETF Protocols

- Overview, RFC

- Internet Drafts

- Draft: syslog-sign

Still Missing

- reliable TCP reconnect

- UI, API and local transport

Analysis

- Classification

- real time monitoring

- Conclusion

Logging

Merriam-Webster Online Dictionary:

log [1, noun]

3 a: the record of the rate of a ship's speed or of her daily progress; also: the full nautical record of a ship's voyage

b: the full record of a flight by an aircraft

4: a record of performance, events, or day-to-day activities

log [2, verb]

2: to make a note or record of: enter details of or about in a log

Using Log Data

- Debugging
- Statistics/Planning
- Accountability for user actions
- Detect ongoing attacks
- Examine security incidents

Syslog vs. Audit

Audit

- mandantory
(no action without log)
- needs trusted system
- low-level
(usually in kernel)
- little semantic
content/meaning
- large amount of data

Syslog

- part of application
- controlled by user
- optional and unreliable
- gets semantic
content/meaning
- (often) human readable
messages

Levels of Abstraction

Audit (10 out of 140 events)

```

header,68,10,recvfrom(2),0,Thu May  8 16:41:55 2008, + 349 msec, \
  subject,-1,root,wheel,root,wheel,763,0,0,0.0.0.0,return, \
  failure : Resource temporarily unavailable,4294967295,trailer,68,
header,68,10,connect(2),0,Thu May  8 16:41:55 2008, + 349 msec, \
  subject,mschuett,sshd,sshd,sshd,sshd,80728,53083,56590,141.89.58.200,return, \
  failure : No such file or directory,4294967295,trailer,68,
header,68,10,connect(2),0,Thu May  8 16:41:55 2008, + 349 msec, \
  subject,mschuett,sshd,sshd,sshd,sshd,80728,53083,56590,141.89.58.200,return, \
  failure : No such file or directory,4294967295,trailer,68,
header,68,10,sendto(2),0,Thu May  8 16:41:55 2008, + 351 msec, \
  subject,mschuett,sshd,sshd,sshd,sshd,80728,53083,56590,141.89.58.200,return, \
  failure : Bad file descriptor,4294967295,trailer,68,
header,96,10,OpenSSH login,0,Thu May  8 16:41:55 2008, + 357 msec, \
  subject,-1,-1,-1,-1,-1,80727,80727,52400,141.89.58.200,text, \
  invalid user name "user",return,failure : No such process,4294967295,trailer,96,
header,68,10,auditon(2) - get audit state,0,Thu May  8 16:41:55 2008, + 357 msec, \
  subject,mschuett,root,wheel,root,wheel,80727,53083,56590,141.89.58.200,return, \
  success,0,trailer,68,
header,68,10,sysctl(3),0,Thu May  8 16:41:59 2008, + 535 msec, \
  subject,-1,root,wheel,root,wheel,853,0,0,0.0.0.0,return, \
  success,0,trailer,68,
header,68,10,sysctl(3),0,Thu May  8 16:41:59 2008, + 535 msec, \
  subject,-1,root,wheel,root,wheel,853,0,0,0.0.0.0,return, \
  success,0,trailer,68,
header,68,10,pipe(2),0,Thu May  8 16:41:59 2008, + 589 msec, \
  subject,mschuett,root,wheel,root,wheel,80729,53083,56590,141.89.58.200,return, \
  success,3,trailer,68,

```

Levels of Abstraction

Syslog

```

2008-05-08T16:41:53.00+02:00 fax sshd[80727]: \
    Invalid user user from 141.89.58.200
2008-05-08T16:41:53.00+02:00 fax sshd[80727]: \
    Failed none for invalid user user from 141.89.58.200 port 52400 ssh2
2008-05-08T16:41:54.00+02:00 fax sshd[80727]: \
    Failed password for invalid user user from 141.89.58.200 port 52400 ssh2
2008-05-08T16:41:55.00+02:00 fax sshd[80727]: \
    Failed password for invalid user user from 141.89.58.200 port 52400 ssh2
2008-05-08T16:41:55.00+02:00 fax sshd[80727]: \
    Failed password for invalid user user from 141.89.58.200 port 52400 ssh2

```

Levels of Abstraction

possible Summary

```
3 fax sshd: auth error \  
    user XY from 141.89.58.200
```

BSD Syslog

- introduced by Eric Allman for sendmail (~ 1984)
- primarily designed for programming/debugging
- simple, uniform, easy to use and configure
- de facto standard for logging on Unix
- 2001: RFC 3164 describes common features

BSD Syslog

- introduced by Eric Allman for sendmail (~ 1984)
- primarily designed for programming/debugging
- simple, uniform, easy to use and configure
- de facto standard for logging on Unix
- 2001: RFC 3164 describes common features

Eric wrote syslogd so that all the logs could be brought to one place and cleanly thrown away at once.

BSD Syslog

Combination of:

- message format
- API
- daemon
- protocol for IPC

Syslog Message Format

```
<38>Mar 17 21:57:57 frodo sshd[701]: Connection from 211.74.5.81 port 5991  
<52>Mar 17 13:54:30 192.168.0.42 printer: paper out
```

- Priority
- Header
 - Timestamp
 - Hostname
- Message
 - Tag
 - Content

By convention:

- Priority not written to logfile
- only printable ASCII
- up to 1024 characters

Metadata: Facility & Severity

Facility:

0. kernel messages
1. user-level messages
2. mail system
3. system daemons
4. security/authorization messages (auth)
5. messages generated internally by syslogd
6. line printer subsystem
7. network news subsystem
8. UUCP subsystem
9. clock daemon
10. security/authorization messages (authpriv)
11. FTP daemon
12. NTP subsystem
13. log audit
14. log alert
15. cron daemon
16. local use 0 (local0)
17. local use 1 (local1)
18. local use 2 (local2)
19. local use 3 (local3)
20. local use 4 (local4)
21. local use 5 (local5)
22. local use 6 (local6)
23. local use 7 (local7)

Severity:

0. Emergency: system is unusable
1. Alert: action must be taken immediately
2. Critical: critical conditions
3. Error: error conditions
4. Warning: warning conditions
5. Notice: normal but significant condition
6. Informational: informational messages
7. Debug: debug-level messages

$$\text{Priority} := 8 \times \text{Facility} + \text{Severity}$$

API

- `#include <syslog.h>`
- `void openlog(const char *ident, \`
 `int logopt, int facility);`
allocate resources and set options
- `int setlogmask(int maskpri);`
set mask to log only severe messages
- `void syslog(int priority, \`
 `const char *message, ...);`
send one log message, arguments like `printf()`
- `void closelog(void);`
free resources

Send Messages

- in programs:

```
#include <syslog.h>
openlog("my_tool", LOG_PID, 0);
setlogmask(LOG_UPTO(LOG_INFO));
syslog(LOG_LPR|LOG_ALERT, "printer on fire");
closelog();
```

- from shell:

```
/usr/bin/logger -t my_tool -i -p lpr.alert \  
                "printer on fire"
```

- by pipe (e. g. in httpd.conf):

```
ErrorLog "|/usr/bin/logger -p local3.info -t apache"
```

syslogd

- collects messages from kernel, applications, and network
- filters by priority
- writes to files, programs, terminals, or network (UDP)
- newer implementations (like syslog-ng or rsyslog) with additional features, like:
 - filter by host/program/regex
 - different message and timestamp formats
 - TCP or SQL servers as destinations

syslog.conf

One rule on every line: Selector <Tab> Action

Example:

```
authpriv.*                /var/log/secure

*.info;mail.none;authpriv.none  -/var/log/messages

auth.*                    |exec /usr/local/sbin/authfilter

*.alert                   root,eric

mail.*                    @logserver.example.org
```

“Transport Protocol”

Input from

- local applications: `socket(AF_UNIX, SOCK_DGRAM, 0);`
- network: `socket(AF_INET, SOCK_DGRAM, 0);`
- kernel: `open("/dev/klog", O_RDONLY, 0);`
(file interface to ring buffer)

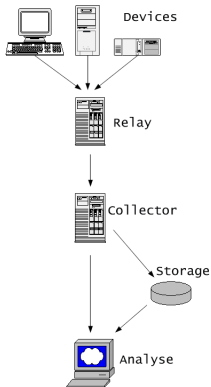
⇒ One message per `recvfrom()/read()`.

Networked Logging

Usual installation: many systems send logs to central log server.

- ease of use
- central backup, monitoring, analysis
- enables event correlation across hosts
- lower risk of unauthorized access and alteration

Roles, Devices, Data Flow



Syslog-Roles:

- *Devices/Sender* generate,
- *Relays* forward,
- *Collectors* receive messages

Further processing:

- *Storage* for archiving
- *Analyzer* to analyze and/or summarize data

Problems with UDP

Advantages:

- simple and efficient (usable for embedded devices)
- allows "stealth logging" (in Honeynets)

Problems:

- packet loss
- no sender authentication

from `man syslogd`:

The ability to log messages received in UDP packets is equivalent to an unauthenticated remote disk-filling service. . .

Move to TCP/TLS

Since ~ 2000:

syslogd implementations support TCP transport.

- no standard established, not always interoperable
- define '\n' or '\0' to mark end of message (to send datagrams in stream)
- possible to tunnel TCP over TLS
- possible to authenticate servers/clients

IETF syslog-sec Working Group

- RFC 3164: The BSD Syslog Protocol (informational)
- RFC 3195: Reliable Delivery for Syslog
- current Drafts:
 - The Syslog Protocol
 - UDP transport mapping for Syslog
 - TLS transport mapping for Syslog
 - Signed Syslog Messages
 - Syslog Management Information Base

RFC 3195: Reliable Delivery for Syslog

- uses BEEP for transport
- RAW-Mode: just BSD Syslog lines as an application/octet-stream payload
- COOKED-Mode: exchange of XML records

```

C: Content-Type: application/beep+xml
C:
C: <entry facility='24' severity='5'
C:   timestamp='Oct 27 13:24:12'
C:   deviceFQDN='screen.lowry.records.example.com'
C:   deviceIP='10.0.0.47'
C:   pathID='173'
C:   tag='dvd'>
C:     Job paused - Boss watching.
C: </entry>
C: END

S: Content-Type: application/beep+xml
S:
S: <ok/>
S: END

```

RFC 3195: Reliable Delivery for Syslog

Known implementations:

- LibLogging (R. Gerhards)
- SDSC Syslog (San Diego Supercomputer Center)

Known users:

- none (?)

Draft: The Syslog Protocol

- keeps plain text format
- allows UTF-8 for data fields
- independent from transport protocol
- full timestamps
- message IDs (like Windows Eventlog)
- structured data fields with namespaces!
(derived from SNMP Private Enterprise Codes)
- message length:
MUST accept 480 octets, SHOULD accept 2048 octets,
MAY receive larger messages (if larger: SHOULD
truncate the payload, MAY discard the message)

Syslog-protocol message format

```
<165>1 2003-10-11T22:14:15.003Z frodo.example.com evnts - ID47
[exampleSDID@0 iut="3" eventID="1011" eventSource="Application"]
```

An application event log entry . . .

- Header
 - Priority
 - Version (*new*)
 - Timestamp (*extended*)
 - Hostname
 - Application Name
 - Process ID
 - Message ID (*new*)
- Structured Data (*new*)
- text Message (*UTF-8*)

Draft: Transmission of syslog messages over UDP

- just like UDP transport in BSD Syslog
- one message per datagram

Draft: TLS Transport Mapping for Syslog

- data encryption, integrity, and host authentication
- requires server and client certificates
- difficult part: mutual authentication using certificate, certificate fingerprint, or CA
- datagram encapsulation:

```
APPLICATION-DATA = 1*SYSLOG-FRAME
SYSLOG-FRAME = MSG-LEN SP SYSLOG-MSG
```

Draft: Syslog Management Information Base

- describes MIB subtree
- allows SNMP monitoring of syslog daemons
- structure:
 - syslogNotifications
 - syslogObjects
 - syslogControlTable (contains configuration parameters like role, address, max. message size)
 - syslogOperationsTable (contains operations information like number of processed messages, start time, time of last error)
 - conformance

Draft: Signed syslog Messages

Overview

- adds detached, in-band signatures
- provides origin authentication, message integrity, replay resistance, message sequencing, and detection of missing messages
- Payload Blocks: contains public key, sent at session/message stream
- Signature Blocks: signed message, contains hashes of last messages

Draft: Signed syslog Messages

Signature Groups

- Problem: Different message destinations
- ⇒ Signature Groups: group messages by message priorities
- Signature Blocks for every Signature Group seperately
- predefined modes:
 - only one global Signature Group
 - each PRI value has its own Signature Group
 - range of sequential PRI values per Signature Group
 - any other (custom) scheme

Draft: Signed syslog Messages

Payload Blocks

on startup:

- Payload Block with
 - timestamp
 - key type (PKIX (X.509), OpenPGP, public key, predistributed, or other/installation-specific)
 - key blob (base64)

- example:

```
2008-05-21T13:47:36.003+0200 C LWQga2V5IGJsb2IK
```

- keys may be large

⇒ Payload Block fragmented into multiple Certificate Blocks

Draft: Signed syslog Messages

Certificate Blocks

- Certificate Blocks with ssign-cert SDs:

```
[ssign-cert VER="xxx" RSID="xxx" SG="xxx" SPRI="xxx"
  TBPL="xxx" INDEX="xxx" FLEN="xxx" FRAG="xxx" SIGN="xxx"]
```

- Version (including hash and signature algorithm)
- Reboot Session ID
- Signature Group
- Signature Priority
- Total Payload Block Length
- Index into Payload Block
- Fragment Length
- Payload Block Fragment (base64)
- Signature (for this CB, base64)

Draft: Signed syslog Messages

Certificate Blocks

example:

```
<110>1 2008-05-21T13:55:00.000+0200 frodo.example.com
- - - [ssign-cert VER="0111" RSID="0"
SG="0" SPRI="0" TBPL="47" INDEX="1" FLEN="47"
FRAG="2008-05-21T13:47:36.003+0200 C LWQga2V5IGJsb2IK"
SIGN="<signature>"]
```

Draft: Signed syslog Messages

Signature Blocks

- Signature Block with with ssign SDs:

```
[ssign VER="xxx" RSID="xxx" SG="xxx" SPRI="xxx"
GBC="xxx" FMN="xxx" CNT="xxx" HB="xxx" SIGN="xxx"]
```

- Version (including hash and signature algorithm)
- Reboot Session ID
- Signature Group
- Signature Priority
- Global Block Counter
- First Message Number
- Count
- Hash Blocks (base64)
- Signature (for this SB, base64)

Problem: TCP unreliable at reconnect

- single TCP connection is reliable, but on close/reconnect a message is lost
- Problem with TCP (and other SOCK_STREAM interfaces?): asynchronous send().
- even after server closes connection: successful send(), writing into local buffer
- known problem, e.g. in *syslog-ng*

Problem: TCP unreliable at reconnect

possible solutions

suggested solution: librelp: Reliable Event Logging Protocol

- one more protocol layer
- client sends messages with transaction numbers
- server acknowledges every message

Problem: TCP unreliable at reconnect

possible solutions

in LANs:

- use SCTP in 1-to-1 modus
- use struct tcp_info to check connection state

```
getsockopt(sock, IPPROTO_TCP, TCP_INFO, &tinfo, &tinfo_size);
```

- check recv() to receive possible status changes

```
rc = recv(sock, msgbuf, BUFSIZE, MSG_DONTWAIT | MSG_PEEK);
```

syslog-protocol vs. syslog-sign

- Structured Data allows adding data in transport
- without impairing later parsing/analysis
- useful with relays or time synchronization problems
- examples:

```
<34>1 1970-01-01T00:01:15.009Z mymachine.example.com
  syslogd - - [transport@0 relay1="logserver.example.com"
  timestamp1="2003-10-11T18:54:16.702+04:00"] restart
<34>1 2003-10-11T14:00:00.000+04:00 mymachine.example.com
  syslogd - - [transport@0 relay1="logserver.example.com"
  timestamp1="2003-10-11T18:20:16.702+04:00"] restart
```

- but: any change invalidates signature

Extended API for syslog-protocol

- syslog(3) API cannot use all syslog-protocol fields
- no MSGID field
- no structured data
- new API desirable
- possible but incomplete approach: Apple's asl(3) API:

```

aslmsg m = asl_new(ASL_TYPE_MSG);
asl_set(m, ASL_KEY_FACILITY, "com.secrets.r.us");

# allows arbitrary key=value pairs, but no namespaces:
asl_set(m, "Clearance", "Top Secret");

...
asl_log(NULL, m, ASL_LEVEL_NOTICE, "Message One");
...
asl_log(NULL, m, ASL_LEVEL_ERR, "Message Two");

```

Problem: syslog(3) and Local Sockets

- unreliable (if buffer full)
- whole message is controlled by the user/application
- thus syslogd depends on user-supplied data
- easy message injection:

```
echo -n "<32>Apr  1 03:14:15 mail sshd[1]: \  
  Accepted publickey for root \  
  from 192.168.23.42 port 1024 ssh2" \  
| socat -u stdin unix-sendto:/var/run/log
```

Problem: syslog(3) and Local Sockets

possible solutions

- device driver to append/inject trusted metadata (timestamp, uid, pid)
- SCM_CREDENTIALS:
 - few systems support passing credentials over local sockets (kernel-verified)
 - `setsockopt(sock, SOL_SOCKET, SO_PASSCRED, &on, sizeof(on));`
 - different implementations and data structures

Problem: Policy for **Reliable** Syslog

Scenario: network problem, logserver unreachable.

Three alternatives:

- throw away messages
- enlarge the buffer (in memory or on disk)
- blocking syslog(3) calls, i.e. stop processes

What to do now?

Introduction

○○
○○○○

BSD Syslog

○○
○○
○○
○○
○○○○

IETF Protocols

○○○
○○○○
○○○○○

Still Missing

○○○
○○○○○

Analysis

○○○○○○○○
○○○○○○○○
○○

Log File Analysis

99,9% IES unimportant

```

Jul 9 09:37:58 localhost SystemStarter: Willkommen!
Jul 9 09:37:58 localhost lookupd[122]: lookupd (version 324.2.1) starting - Fri Jul 9 09:37:58 2004
Jul 9 09:37:58 localhost ConsoleMessage: Starting Apple Multicast DNS Responder
Jul 9 09:37:58 localhost ConsoleMessage: Starting kernel event agent
Jul 9 09:37:58 localhost ConsoleMessage: Starting timed execution services
Jul 9 09:37:58 localhost ConsoleMessage: Starting SecurityServer
Jul 9 09:38:01 localhost SystemStarter: ?Apple Multicast DNS Responder? wird gestartet
Jul 9 09:38:02 localhost SystemStarter: ?Kernel Event Agent? wird gestartet
Jul 9 09:38:02 localhost SystemStarter: ?Timed Execution Services? werden gestartet
Jul 9 09:38:04 localhost kernel: ApplePMUUserClient::setProperties WakeOnACChange 0
Jul 9 09:38:06 localhost ConsoleMessage: Initializing network
Jul 9 09:38:06 localhost mDNSResponder[155]: mDNSResponder-58.8 (Apr 24 2004 20:38:40) starting
Jul 9 09:38:06 localhost SystemStarter: Starting SecurityServer
Jul 9 09:38:06 localhost SystemStarter: Netzwerkdienste werden initialisiert
Jul 9 09:38:06 localhost ConsoleMessage: Checking disks
Jul 9 09:38:06 localhost SystemStarter: Volumes werden ?berprft
Jul 9 09:38:08 localhost kernel: ATY,Be_A: vram [9c000000:02000000]
Jul 9 09:38:08 localhost kernel: ATY,Be_B: vram [98000000:02000000]
Jul 9 09:38:08 localhost syslogd: /dev/console: Input/output error
Jul 9 09:38:08 localhost init: kernel security level changed from 0 to 1
Jul 9 09:38:09 localhost DirectoryService[186]: Launched version 1.8 (v256.6)
Jul 9 09:38:09 localhost loginwindow[182]: Sent launch request message to DirectoryService mach_init port
Jul 9 09:38:16 localhost SystemStarter: Warten auf ?Netzwerkdienste werden initialisiert?
Jul 9 09:38:22 localhost last message repeated 2 times
Jul 9 09:38:23 localhost config[87]: executing /System/Library/SystemConfiguration/Kicker.bundle/Contents/Resources/set-hostname
Jul 9 09:38:24 localhost set-hostname[195]: setting hostname to ivich.local
Jul 9 09:38:25 localhost SystemStarter: Warten auf ?Netzwerkdienste werden initialisiert?
Jul 9 09:38:28 localhost ConsoleMessage: Loading Shared IP extension
Jul 9 09:38:28 localhost SystemStarter: Shared-IP-Erweiterung wird geladen
Jul 9 09:38:28 localhost /usr/libexec/crashreporterd: get_exception_ports() failed: (ipc/send) invalid destination port
Jul 9 09:38:28 localhost ConsoleMessage: Starting Apple File Service
Jul 9 09:38:28 localhost ConsoleMessage: Starting printing services
Jul 9 09:38:29 localhost SystemStarter: ?Apple File Services? werden gestartet
Jul 9 09:38:29 localhost SystemStarter: Die Druckerdienste werden gestartet
Jul 9 09:38:30 localhost ConsoleMessage: Loading IP Firewall extension
Jul 9 09:38:30 localhost SystemStarter: IP-Firewall-Erweiterung wird geladen
Jul 9 09:38:32 localhost kernel: IP packet filtering initialized, divert enabled, rule-based forwarding enabled, default to accept, logging disabled
Jul 9 09:38:32 localhost kernel: IPv6 packet filtering initialized, default to accept, logging disabled
Jul 9 09:38:32 localhost kernel: IP firewall loaded
Jul 9 09:38:32 localhost ConsoleMessage: Starting internet services
Jul 9 09:38:32 localhost SystemStarter: ?Internet Services? werden gestartet
Jul 9 09:38:32 localhost xinetd[257]: xinetd Version 2.3.11 started with libwrap options compiled in.
Jul 9 09:38:32 localhost xinetd[257]: Started working: 3 available services
Jul 9 09:38:33 localhost SystemStarter: Systemstart beendet.
Jul 9 09:38:44 localhost diskarbitrationd[88]: disk1s2 hfs ED021A74-9EEF-3AA5-9B54-5B703F7D0D71 ute [not mounted]
Jul 9 09:38:45 localhost diskarbitrationd[88]: disk1s2 hfs ED021A74-9EEF-3AA5-9B54-5B703F7D0D71 ute /Users/ute

```

Normalization

- messages from different sources in different formats (Syslog, multilog, Windows Eventlog, SNMP traps, ...)
- convert into common format
 - timestamp format
 - character set
 - data fields

Example: Eventlog → Syslog

```
2008-05-06T15:18:01.00+02:00 THESEUS NT: <NTSYSLOG;I2;> \
  Service 'NTSYSLOG' wurde gestartet
2008-05-06T15:17:58.00+02:00 THESEUS NT: <EventLog;I6005;> \
  Der Ereignisprotokolldienst wurde gestartet.
```

grep

- `grep "important Msg" logdatei | less`
- `grep -v "unimportant Msg1" logdatei | \
grep -v "unimportant Msg2" | \
grep -v "unimportant Msg3" | less`
- inefficient
- only interactive

grep and Shell Tools

```
cat logfile \  
| egrep '^.{15} .+ sshd\[.+\]: Invalid user .+ from .+$' \  
| sed -e 's/^.\{16\}//' -e 's/ sshd\[.*\]: / sshd: /' \  
-e 's/user .* from/user XY from/' \  
| sort | uniq -c | sort -nr >> report
```

Ergebnis:

Summary: lines.login_failed_user

```
-----  
451 neo sshd: Invalid user XY from 210.75.23.122  
451 mail sshd: Invalid user XY from 210.75.23.122  
209 neo sshd: Invalid user XY from 60.191.41.89  
12 neo sshd: Invalid user XY from 125.152.17.236  
12 mail sshd: Invalid user XY from 125.152.17.236
```

Artificial Ignorance

- filter known good messages
- leave new and unknown messages for inspection

Steps:

1. take existing logs, strip timestamps & PIDs
2. list with `sort | uniq -c`
3. write regexps to filter known good messages
4. regularly review remaining messages
5. readjust filters

logcheck

- uses lists of regexps to classify:
 - hacking
 - violations
 - ignore
 - everything else
- sends report by e-mail
- uses logtail to remember current position in logfile
- can be called as often as necessary (hourly, daily, weekly)

logwatch

- similar to logcheck
- calls Perl scripts for service-dependent summary
- e.g. for amavisd:

```

***** Summary *****
      2  Miscellaneous warnings
=====
19664  Ham ----- 95.36%
19630  Clean passed 95.19%
   34  Bad header passed 0.16%

   942  Spam ----- 4.57%
   514  Spam blocked 2.49%
   428  Spam discarded (no quarantine) 2.08%

    15  Malware ----- 0.07%
    15  Malware blocked 0.07%

 20621  Total messages scanned ----- 100.00%
662.993M  Total bytes scanned 695,198,092
=====

```

service-specific summaries

e. g. pfllogsumm for Postfix

Postfix log summaries for May 21

Grand Totals

messages

```

7708 received
27350 delivered
  3 forwarded
  70 deferred (575 deferrals)
240 bounced
48996 rejected (63%)
  0 reject warnings
  0 held
262 discarded (0%)

```

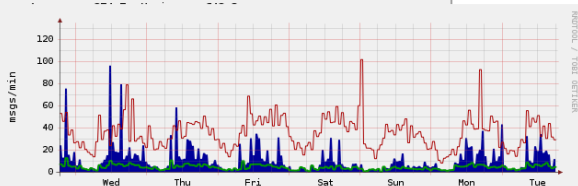
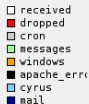
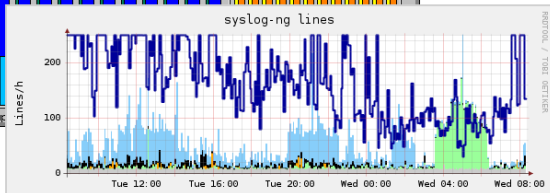
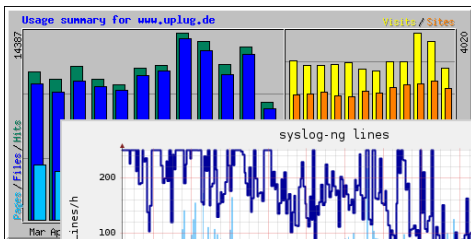
```

1024m bytes received
4036m bytes delivered
 931 senders
 520 sending hosts/domains
3621 recipients
 506 recipient hosts/domains

```

...

Counting Events



	total:	119521	msgs	avg:	11.73	msgs/min	max:	3480	msgs/min
■ Sent	total:	45801	msgs	avg:	4.50	msgs/min	max:	79	msgs/min
■ Received	total:	362697	msgs	avg:	35.69	msgs/min	max:	237	msgs/min
■ Rejected									

real time monitoring

advantages:

- follow sequences and dependencies of related messages
- stateful analysis (e.g. follow user session)
- automatic reactions (notification or countermeasure)

some programs:

- special purpose:
 - Snort (network IDS)
 - Nagios (device/service monitoring)
 - fail2ban (ssh logins)
- general purpose:
 - swatch
 - logsurfer
 - Simple Event Correlator

Swatch

- matches lines against regexp
- performs actions on match

swatch.conf example:

```
ignore /ntpd.*: kernel time sync enabled/

watchfor /[fF]ailed password for (.*?) from (.*?) port (.*?)$/
    throttle threshold=3,delay=0:1:0,key=$2
    exec "/sbin/iptables -A swatch_rejects -s $2 -j DROP"
```

Logsurfer/Logsurfer+

- matches lines against `match-regex` and `not-match-regex`
- performs usual actions
- feature: create/delete rules
- collect multiple lines in a context
- complex configuration with multiple levels of quoting

rule format:

```
match-regex not-match-regex  
  stop-regex not-stop-regex timeout [continue] action
```

Logsurfer/Logsurfer+

advanced logsurfer.conf example:

```
'on (/.*): file system full$' '/root/mp3sammlung' - - 0
    exec "/usr/bin/find" "$1 -name *.mp3 -exec rm {} ;"

'failed password for (.*?) from (.*?) port (.*?)$' - - - 0
    pipe "report.sh notice"

'.*' - - - 0
    echo >>lines.unknown $0
```

contexts

```
'ftpd\[ [0-9]+\]: \(\?@\([0-9\.]+\)\) \[INFO\] Neue Verbindung von \1\.$' - - - 0
  open
  "ftpd\[ [0-9]+\]: \(.*\@$2\)" - 2000 720 20 ignore

'ftpd\[ [0-9]+\]: \(.*\@(.*\) \) \[ERROR\]' - - - 0
  rule before
  "ftpd\\\[[0-9]+\:\\\]: \\(\.*\@(\$2)\\) \\\[INFO\\] Logout." - - - 300
    report "/bin/cat" "ftpd\\\[[0-9]+\]\\: \(\.\\.*\@$2\)"
```

created context:

```
ftpd\[ [0-9]+\]: (.*\@192.168.0.2)
```

created rule:

```
ftpd\[ [0-9]+\]: \(.*\@(192.168.0.2)\) \[INFO\] Logout. -
- - 300
  REPORT "/bin/cat" "ftpd\[ [0-9]+\]: \(\.\\.*\@$2\)"
```

Simple Event Correlator

- can count events and observe thresholds
- dynamic creation/deletion of rules
- create own events
- arbitrary context names
- execute custom Perl functions

Simple Event Correlator

sec.conf example:

```

type=SingleWith2Thresholds
ptype=RegExp
pattern=sshd.*: authentication failure[ ;].* rhost=(\S+)

desc=Multiple failed ssh authentication attempts from $1
action=logonly ; shellcmd ( /usr/local/sbin/iptables-add [...] )
window=60
thresh=4

desc2=Pruning iptables firewall rule blocking ssh from $1
action2=logonly ; shellcmd ( /sbin/iptables [...] )
window2=7200
thresh2=0

```

note on automatic actions

- possible notifications:
SNMP-traps, e-mail, Jabber, SMS, ...
- must not cause new errors
- countermeasures only in known environment
- log messages are user input – always quote and filter in scripts

```
sshd[164]: ... illegal user 'rm -rf *'
```

recommendations

- use only few logfiles (not every facility separate)
- automate, filter, summarize
- read only new or known bad messages yourself, but do read them
- put analysis/summary in periodic daily
- do not plan overly complex (classification is good, time-based thresholds are nice to have, dynamic rules and contexts are seldom necessary)

Links: Information

- NetBSD GSoC08 project: syslogd
- IETF Syslog Working Group Status Page
- LogAnalysis.org
- Syslog-ng FAQ und Loghost HOWTO

Links: Daemons

- rsyslog
- syslog-ng
- stunnel

Links: Analysis

- logcheck
- logwatch
- Swatch
- LoGS
- SEC - simple event correlator
- Logsurfer
- Logsurfer+
- Artificial Ignorance: How-To Guide