On System Administration:

We seem to be made to suffer. It's our lot in life.

--C-3PO  Star Wars (1977)
Log Analysis:
cat /var/log/messages | grep -v ... | grep -v ... | grep -v ... | grep -v ... | grep -v ...
“Real-Time” Log Analysis:

tail -f /var/log/messages | grep -v ... | grep -v ... | grep -v ... | grep -v ...
LoGS History:

- “Thou Shalt analyze thine logfiles...” --NCSA 2000-2001
- ... “With SWATCH”
- “... Give it to the new guy” -- my (then) co-workers
- Working with Logsurfer in 2001-2004
- Started discussing LoGS Jan/Feb 2003
- Began LoGS in June 2003
What’s wrong with Logsurfer?

- Syntax can be less than clear ala:
  - `.*` - - - 0 exec "/bin/echo $0"
- Too much fork()-ing around! (Logsurfer+ helps a little)
- Traversing a linear ruleset is less-than efficient
- Dynamic rules can get hairy:
  - rules that create rules ... escaping horrors!
- No way to extend Logsurfer (without hacking the source)
- A “real” programming language would be nice
Whats wrong with Logsurfer? (cont)

- Regular expressions are nice, but...
- Compiling is pretty cool!
- CL-PPRE is the fastest regex engine out there...
  - suck it, Perl! (C is slow! ;)
- ruleset language is hard to write ‘maintainable rulesets’
  - Kludges using M4 (and similar) “macro processors”
Maintainability?

divert(0)dnl
# Sendmail MTA
divert(-1)dnl
define(`HAVE_SENDMAIL')dnl
define(`ONE', `undivert(1)')dnl
define(`TWO', `undivert(2)')dnl
divert(1)dnl
'sendmail' - - - 0 ignore
divert(2)dnl
'\.\{16}\(.*\) sendmail\[[0-9]+\]: My unqualified host name.*unknown; sleeping for retry' - - - 0 ignore
'\.\{16\}(.*) sendmail\[[0-9]-\]+\: unable to qualify my own domain name.*using short name' - - - 0 ignore
'\.\{16\}(.*) sendmail\[[0-9]+\]: gethostbyaddr\(.*\) failed:' - - - 0 ignore
'\.\{16\}(.*) sendmail\[[0-9]+\]: rejecting connections on daemon (.*): load average: (.*)' - - - 0
  exec "/usr/local/bin/surf_GenericMsg -k logsurfer -h ````$2'''' -s 5 -m "sendmail $3 rejecting connections; load average $4 too high!
"sendmail timeout waiting for input from $3 during $4"
On Lisp:

"This is the weapon of a Jedi Knight. Not as clumsy or as random as a blaster, but an elegant weapon for a more civilized age."

--Obi-Wan
What is LoGS?

- I know you want to read ‘messages’ sequentially from a ‘data source’ and do (one or more) things in response to finding interesting messages

- I don’t know how to find an interesting message

- I don’t know what to do in response

  - I’ve got some guesses that may help you...

  - You can hack it when my guesses are wrong.
LoGS Components

- Messages
- Rules
- Rulesets
- Contexts
- Windows
- Actions

These are all CLOS Objects

(Actions are just Lisp Functions)
What does this look like?

;;; create a rule that prints every message
(make-instance 'rule
  :match
  ;; match every message
  (lambda (message) t)
  :actions
  (list
    ;; print the message
    (lambda (message environment)
      (format t "~A%" (message message)))))

*declarations are good, but make slides harder to read
OMG THATS UGLY!!!
(how about “the RDL” instead?)

(rule
  matching
  (lambda (message) t)
  doing
  (lambda (message)
    (format t “~A~%” (message message))))
On Lisp’s Syntax:

"No different. Different only in your mind. You must unlearn what you have learned"

--Yoda Star Wars Episode V: The Empire Strikes Back (1980)
RDL is “under construction”

- Steal liberally from LOOP
  - Montpellier France, Fresh Air, Red Wine, & Practical Common Lisp
- Make RDL very similar to English
- Lots of stuff left to add!
- Make it hackable! (like “old-school” LOOP)
- I also don’t know how you want to express your rulesets :P
- RDL is a half-finished guess
LoGS Rulesets

- Rulesets are:
  - A doubly linked list containing:
    - Rules and or Linked lists containing:
      - ...
  - Rulesets may contain other rulesets
  - This leads us to a “tree” sort of shape for our ruleset
  - Compare the performance with Logsurfer’s linear list
Worst-Case Scenario:
The last rule matches

- Logsurfer config:
  - 117 regular expressions tested before match found

- LoGS config to implement “identical” ruleset:
  - 13 functions ran with message as input

=> Trees good!
Dynamically Created Rules

Rules may be:
* Created
* Deleted
* Modified

while LoGS is running
LoGS Contexts

- Contexts are known by their names
- LoGS Contexts are *PASSIVE* Message stores
  - In Logsurfer, they are *ACTIVE* message stores
- This leads us to a linear list of contexts that must be queried for each line of input
  - There is no short circuiting!!!!!
- LoGS contexts are similar to SEC’s Contexts
Basic Context Usage

(make-instance 'rule
  :match '#’match-all-messages
  :actions
  (list
    (lambda (message env)
      (add-to-context
        (ensure-context
          :name ‘all-messages)
        message))))
On the Learning Curve of LoGS:

Listen, I can't get involved. I've got work to do. It's not that I like the Empire; I hate it. But there's nothing I can do about it right now... It's all such a long way from here.

--Luke Star Wars (1977)
What is wrong with (just) regular expressions?

- You cannot correlate information not in the log stream*
- correlate errors with system maintenance periods
- find time synchronization problems
- correlate batch scheduler information...

- They can be less than clear

*I am aware that Perl Compatible Regular Expressions are Turing Complete - I’m just not /that/ twisted
Why an ‘Action List’?

- I often want 2 or more things done in response to a given message
  - Mark the node ‘offline’
  - AND send the admin an email
- Conceptually, 2 rules to implement the above feels icky
Lisp is cool!

You can put this in your ruleset file:

```lisp
(defun method get-line :AFTER ((ff file-follower))
  (with-open-file
    (f "./var/run/LoGS/f.pos"
      :direction :output :if-exists :rename)
    (format f "~A~%" (file-position (filestream ff))))
```
Immediate Future:

- More stuff in the RDL
- Better SBCL support
  - CMUCL is best supported today
    - CMUCL does not work on my Intel Mac!
- LoGS binary
  - ./LoGS --ruleset /path/to/ruleset.lisp ...
Further Future:

- “Real” Thread support
- consensus on use of SPECIAL variables
  - Is dynamic scoping inherently evil?
- 2 forks in LoGS CVS tree
- Support for more Lisp implementations
  - LoGS support for many (non-CMUCL, non-SBCL) has crumbled...
Recent Developments:
On Lisp Being an “Old” language:
"When nine hundred years old you reach, look as good you will not."

--Yoda  Star Wars Episode VI: Return of the Jedi (1983)