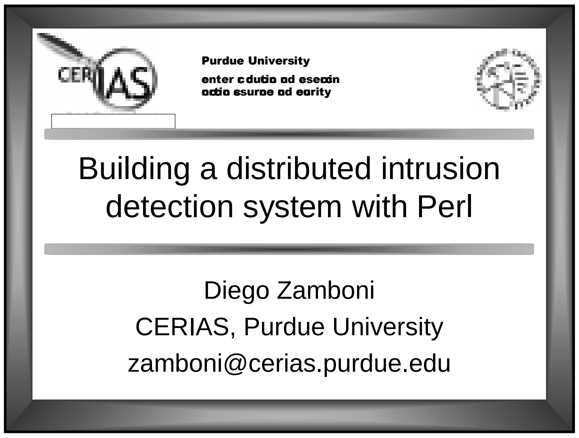
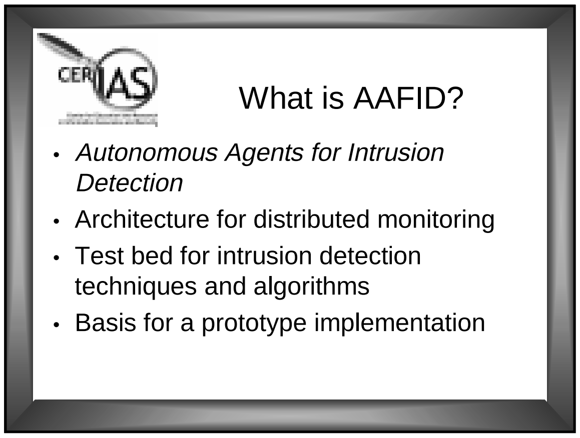
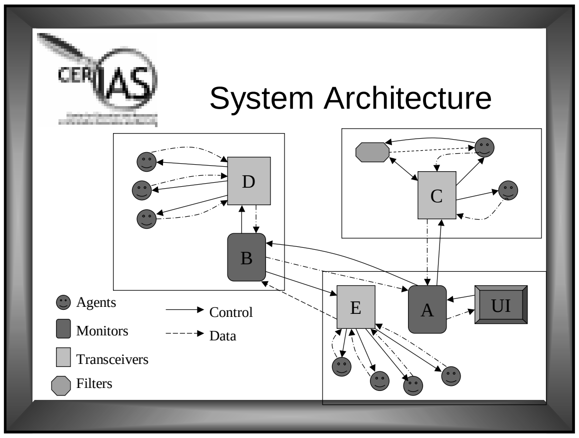

Purdue University
enter cdutio od esecin
octio esurce od eority

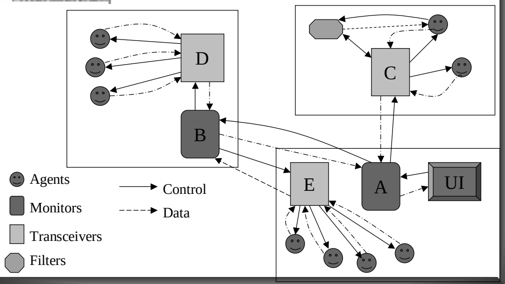
Building a distributed intrusion detection system with Perl

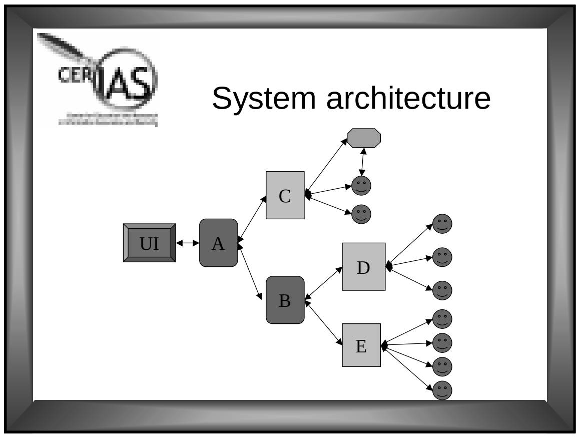
Diego Zamboni
CERIAS, Purdue University
zamboni@cerias.purdue.edu

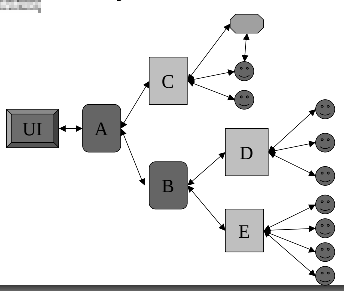

What is AAFID?

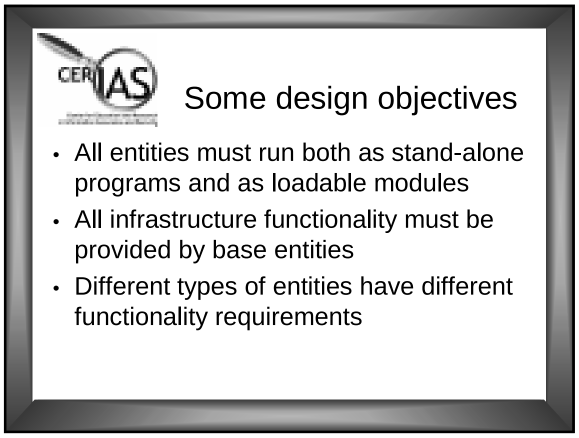
- *Autonomous Agents for Intrusion Detection*
- Architecture for distributed monitoring
- Test bed for intrusion detection techniques and algorithms
- Basis for a prototype implementation


System Architecture

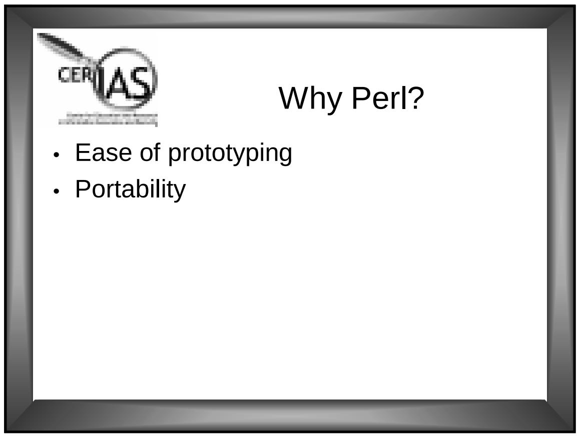



System architecture



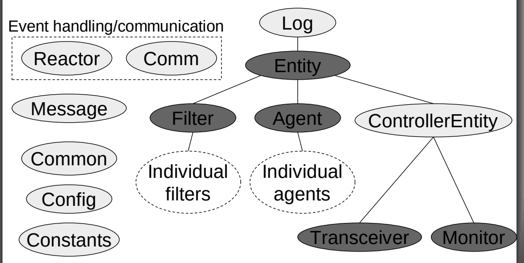

Some design objectives

- All entities must run both as stand-alone programs and as loadable modules
- All infrastructure functionality must be provided by base entities
- Different types of entities have different functionality requirements


Why Perl?

- Ease of prototyping
- Portability

Our object hierarchy (AAFID::)



Event handler

- Comm::Reactor implements a general event handler
- Can react to file, time and signal events
- Arbitrary callbacks (code refs)
- Implemented using IO::Select
- Using class methods instead of instance methods caused some nasty bugs

Cool uses of Perl #1: defining new commands

- Entities react to commands
- Command CMD is defined by a subroutine called **`command_CMD`**
- New commands can be added with very little effort just by defining the appropriate subroutines

Cool uses of Perl #2: named parameters

- Entity objects are implemened with hashes
- Entity parameters are stored as elements in that hash
- Each entity is tied to a hash to allow easy access to parameters (**`$Params{param}`** instead of **`$self->getParameter('param')`**)

Cool uses of Perl #3: hash syntax

- Allows having a very general “data” field in AAFID messages:
  ```
  command add_fs … FS=>”/”, Limit=>85
  ```
- Data::Dumper and eval do all the work for generating and interpreting data fields
- Eval: potential security problems

Cool uses of Perl #4: code generation tool

- Reads a description file, writes Perl code
- Inserts **`# line “file”`** comments to produce meaningful error messages
- Allows definition of new commands with named parameters

A very simple agent

```
NAME: CheckRoot
AUTHOR: Diego Zamboni
DESCRIPTION: Check root dir permissions
VERSION: 0.1
PERIOD: 10
CHECK:
  if (-w “/”) {
     return(10,”Root dir is writable”);
  else {
     return(0,”Everything ok”);
  }
```

Communication mechanisms

- Transceiver-agent: Unix pipes
- Monitor-transceiver: TCP

- Both are transparently used as IO::Handles (at least in Unix)
- All communications are encapsulated, so they are easy to replace or upgrade

Other aspects

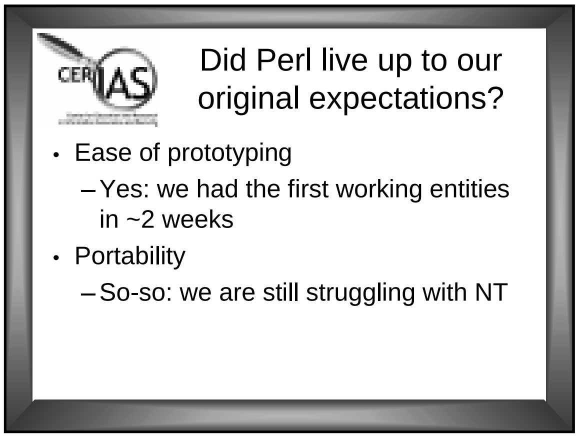
- Graphical User Interface
  – Uses Tk package
  – Very early stages
  – Subject for a lot of future research

AAFID GUI

AAFID GUI

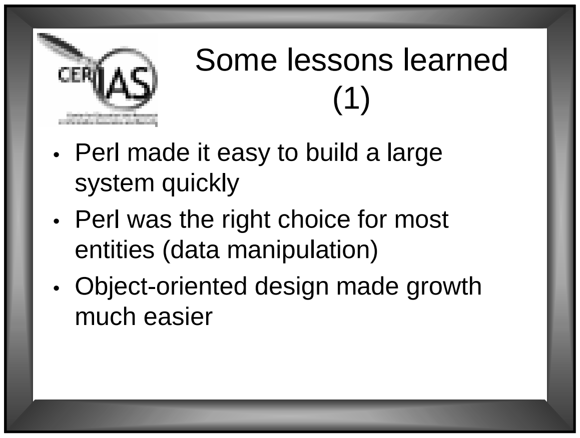
Some modules we used

- IO::{Handle,Select,Socket,File}
- Data::Dumper
- Resources
- Log::Topics
- Tk



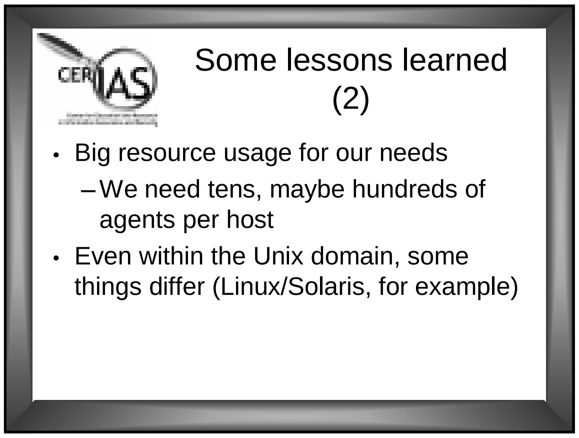
Did Perl live up to our original expectations?

- Ease of prototyping
  - Yes: we had the first working entities in ~2 weeks
- Portability
  - So-so: we are still struggling with NT



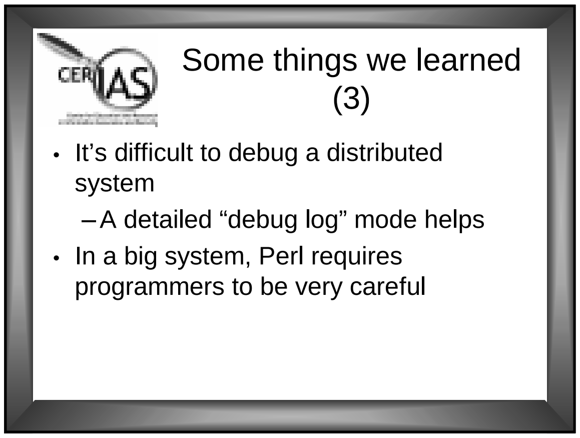
Some lessons learned (1)

- Perl made it easy to build a large system quickly
- Perl was the right choice for most entities (data manipulation)
- Object-oriented design made growth much easier



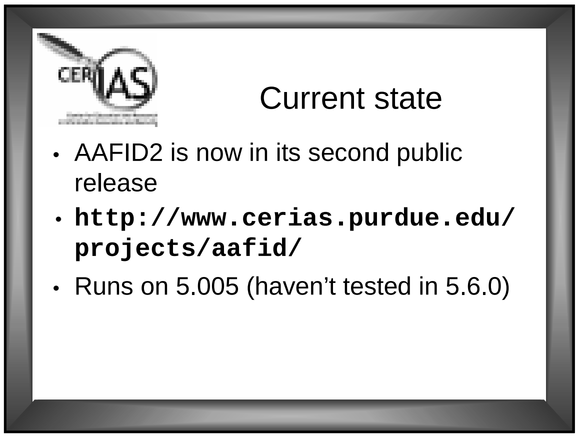
Some lessons learned (2)

- Big resource usage for our needs
  - We need tens, maybe hundreds of agents per host
- Even within the Unix domain, some things differ (Linux/Solaris, for example)



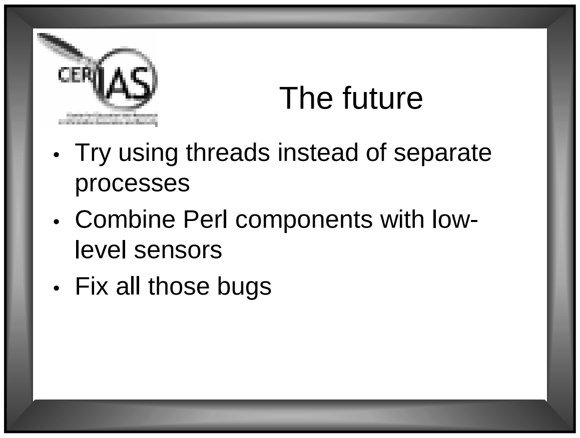
Some things we learned (3)

- It's difficult to debug a distributed system
  - A detailed "debug log" mode helps
- In a big system, Perl requires programmers to be very careful



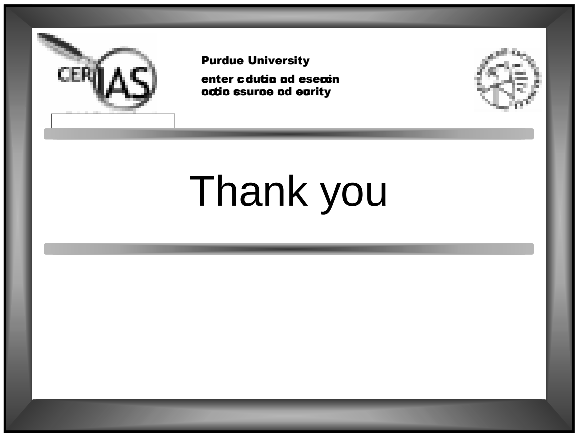
Current state

- AAFID2 is now in its second public release
- **`http://www.cerias.purdue.edu/projects/aafid/`**
- Runs on 5.005 (haven't tested in 5.6.0)



The future

- Try using threads instead of separate processes
- Combine Perl components with low-level sensors
- Fix all those bugs


Purdue University

enter cdutio od esecin
odio esurce od earity

Thank you