SWINBURNE UNIVERSITY OF TECHNOLOGY

DIstributed Firewall and Flow-shaper Using Statistical Evidence (DIFFUSE)

Sebastian Zander,
Grenville Armitage

{szander, garmitage}@swin.edu.au

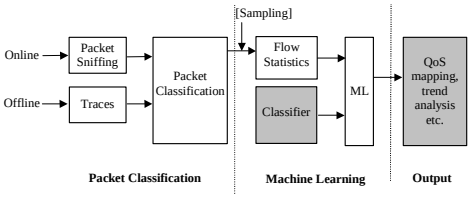
Centre for Advanced Internet Architectures (CAIA)
Swinburne University of Technology

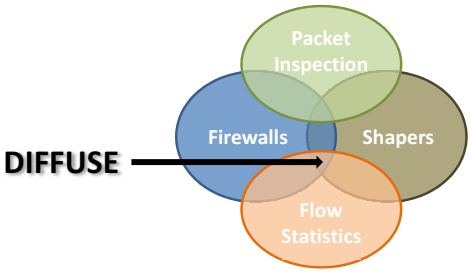
Acknowledgements

- The DIFFUSE project has been made possible in part by a grant from the Cisco University Research Program Fund at Community Foundation Silicon Valley.

Overview

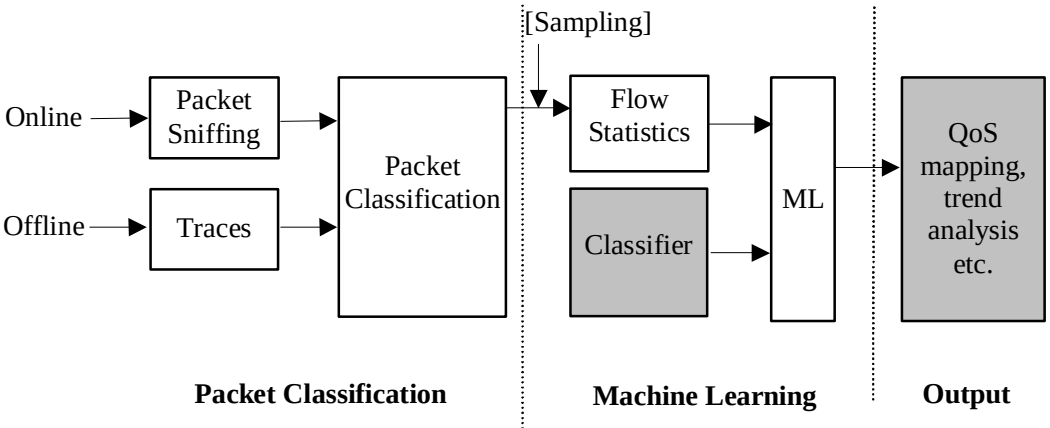
- Machine Learning (ML) traffic classification
- DIFFUSE design and implementation
- Conclusions and future work





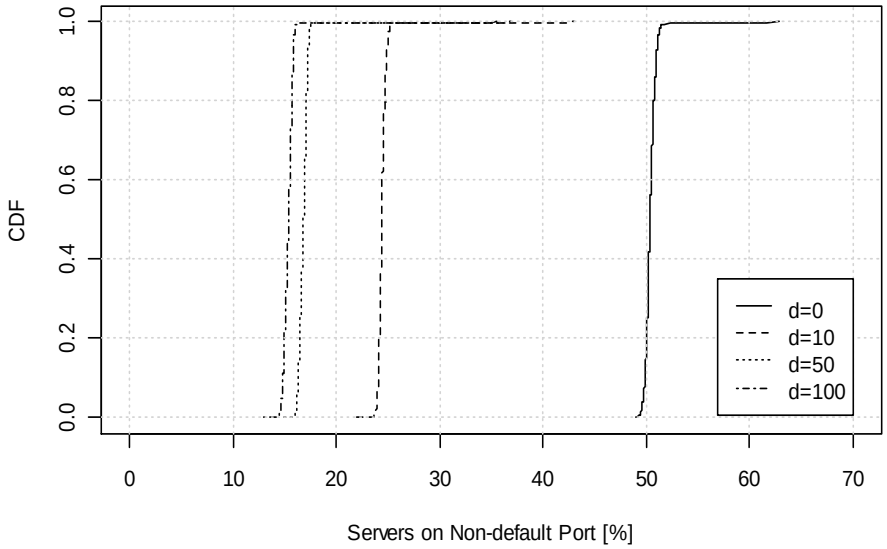
Background and Terminology

- Packets classified into flows
- Flows characterised by statistics (features), e. g. packet length
- Supervised ML techniques build classifier from labelled examples in training phase
- Classifier used to classify unlabelled instances in testing phase



Why ML-based traffic classification?

- Port-based classification does not work well
  - Applications without default ports
  - Default ports not used (NATs, proxies, deliberately)
- Payload inspection is limited
  - No payload, payload encrypted, privacy restrictions
  - Costly to develop signatures (many unsupported protocols)

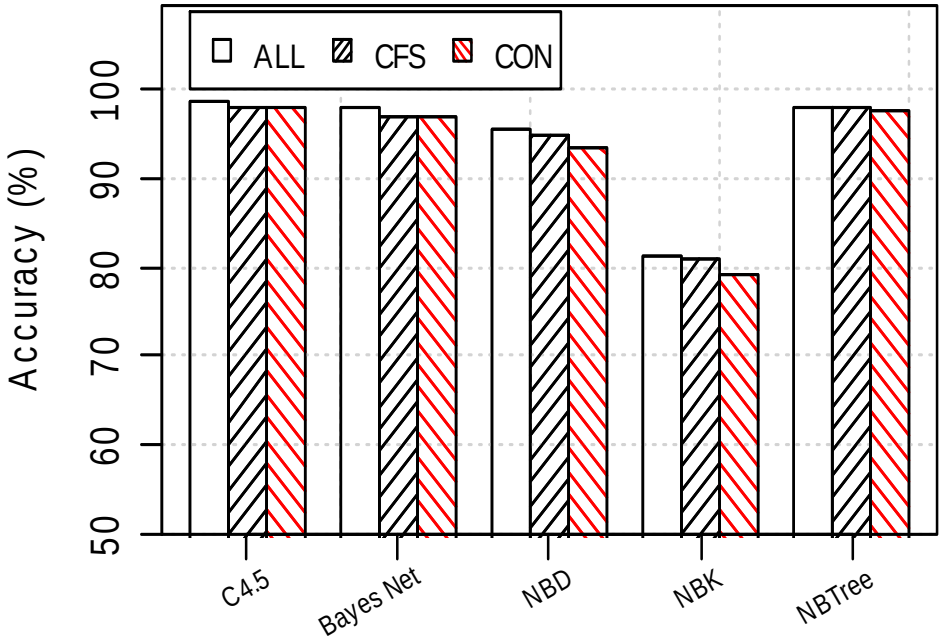


Traffic Classification with Supervised ML

- FTP, Telnet, SMTP, DNS, HTTP, Half-Life
- Naïve Bayes, Bayes Net, Naïve Bayes Tree, C4.5, Nearest Neighbour, Neural Networks, Support Vector Machines
- Correlation-based (CFS), consistency-based (CON) feature selection
- Simple features based on packet length, inter-arrival time, flow duration, packets, bytes
- N. Williams, S. Zander, G. Armitage, "A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification", SIGCOMM CCR, Volume 36, October 2006.
- N. Williams, S. Zander, G. Armitage, "Evaluating Machine Learning Algorithms for Automated Network Application Identification", CAIA TR 060410B, April 2006.

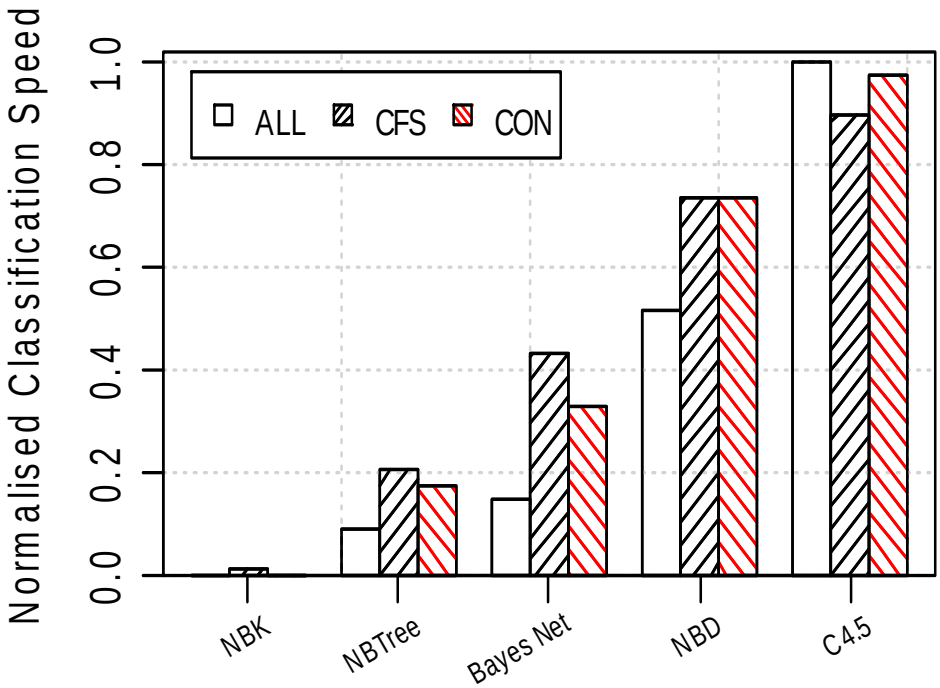
Traffic Classification with Supervised ML

- Better algorithms provide 97–99% accuracy
- CFS/CON reduces learning time with −1–2% accuracy
- Packet length features most useful



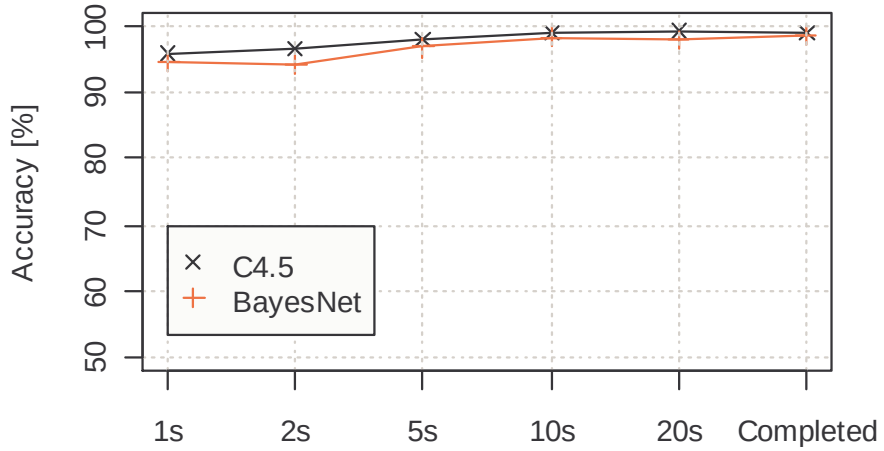
Traffic Classification with Supervised ML

- Very different training/classification speeds
- C4.5 best accuracy, classification speed with acceptable learning time



Game Traffic Classification

- Half Life 1/2, Counter-Strike, Quake 3, Enemy Territory (ET), Halo 2 on Xbox, non-game traffic
- Naïve Bayes, Bayes Net, C4.5 algorithms
- Classify game traffic with $\geq$ 99% accuracy
- Close to maximum accuracy in 5–10 s from flow start
- N. Williams, S. Zander, G. Armitage, "Evaluating Machine Learning Methods for Online Game Traffic Identification", CAIA TR 060410C, April 2006.



Online Classification

- Previous work used features computed for whole flows or initial parts of flows (from start)
- For online classification of long-lasting flows can't wait for flow end
- And what if we miss the start of long-lasting flows?
- Compute features over sliding windows of packets (sub flows)
- For ET game traffic accuracy of 98–99% for 25 packet windows when training on multiple sub flows
- T.T.T. Nguyen, G. Armitage, "Training on multiple sub-flows to optimise the use of Machine Learning classifiers in real-world IP networks", IEEE LCN, November 2006.

Classification of Skype and BitTorrent

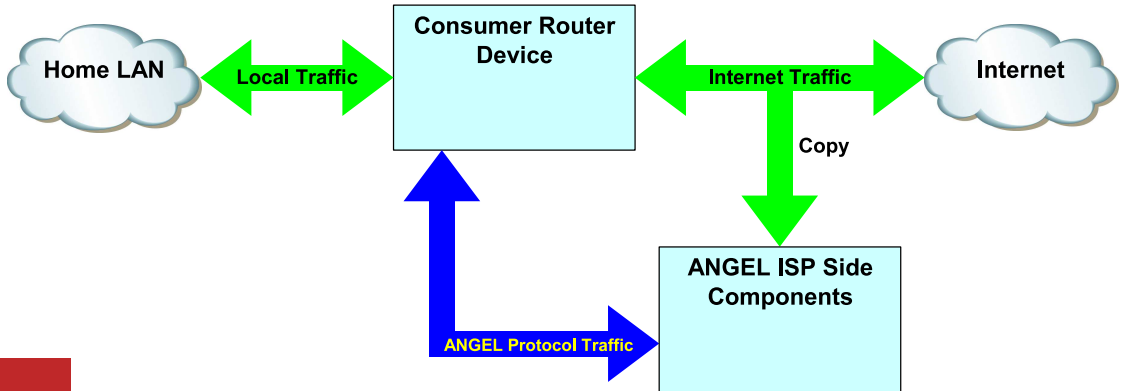
- So far we used simple non-application-specific features only
- Increase classification accuracy with tailored features
- Recent work developed tailored efficient features to detect Skype and BitTorrent with accuracies of 98–99%
- P. Branch, A. Heyde, G. Armitage, "Rapid Identification of Skype Traffic", ACM NOSSDAV 2009, June 2009.
- J. But, P. Branch, T. Le, "Rapid Identification of Bittorrent Traffic", IEEE LCN 2010, October 2010.

What else?

- Not much work in the area when we started in 2004...
- ... Then it suddenly became quite popular
- T.T.T. Nguyen, G. Armitage, "A Survey of Techniques for Internet Traffic Classification using Machine Learning", IEEE Communications Surveys & Tutorials, vol. 10 no. 4 pp. 56-76, 2008.

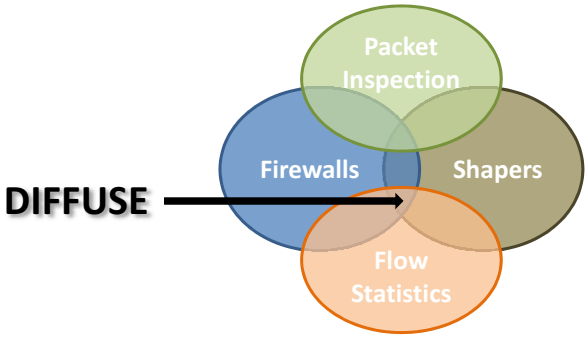
Automated Prioritisation of Interactive Traffic

- Use ML to classify interactive traffic in ISP network
- Dynamically reconfigure ISP network to improve QoS
- Prototype developed in Smart Internet CRC: Automated Network Games Enhancement Layer (ANGEL)
- J. But, N. Williams, S. Zander, L. Stewart, G. Armitage, "ANGEL - Automated Network Games Enhancement Layer", NetGames 2006, November 2006.
- http://caia.swin.edu.au/sitcrc/angel/



DIFFUSE Project

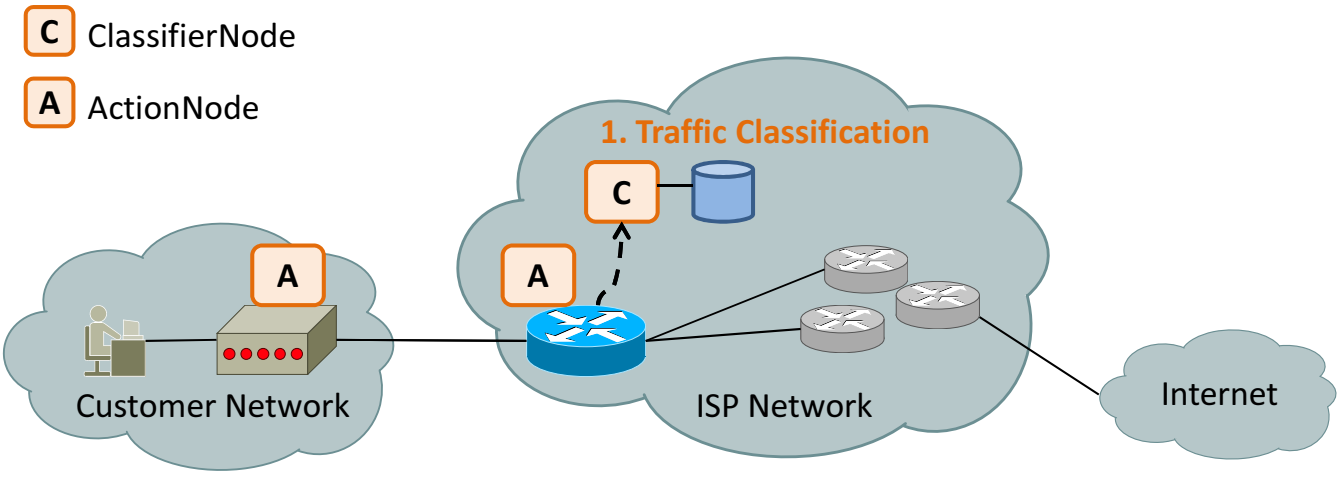
- 12 month Cisco-funded "successor" of ANGEL
- Use features and ML to classify network traffic
- Spatially separate flow classification from flow treatment (e.g. blocking, shaping)
- Applications
  - Automated provision of QoS for interactive traffic
  - Lawful Interception
  - Detection and blocking malicious traffic



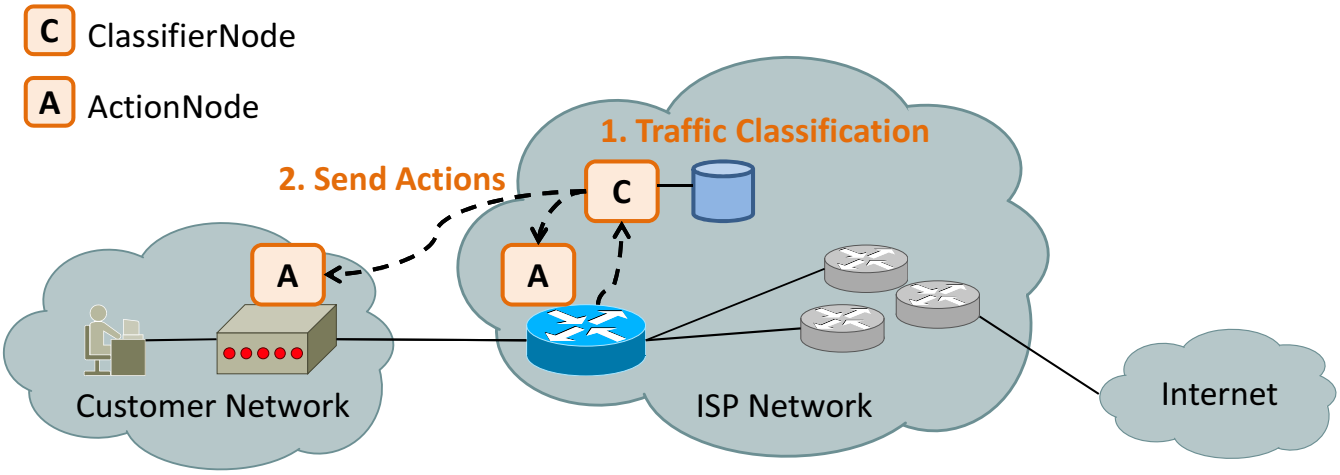
Differences to ANGEL

- Simpler architecture but with explicit extension hooks
- Integration with FreeBSD firewall
  - Higher speed
  - Familiar rule syntax
  - Wide distribution
  - edit gja: FreeBSD shall rule the world!
- Usable for users/admins but also for researchers
- Not limited to QoS for interactive flows
- Research beyond building prototype

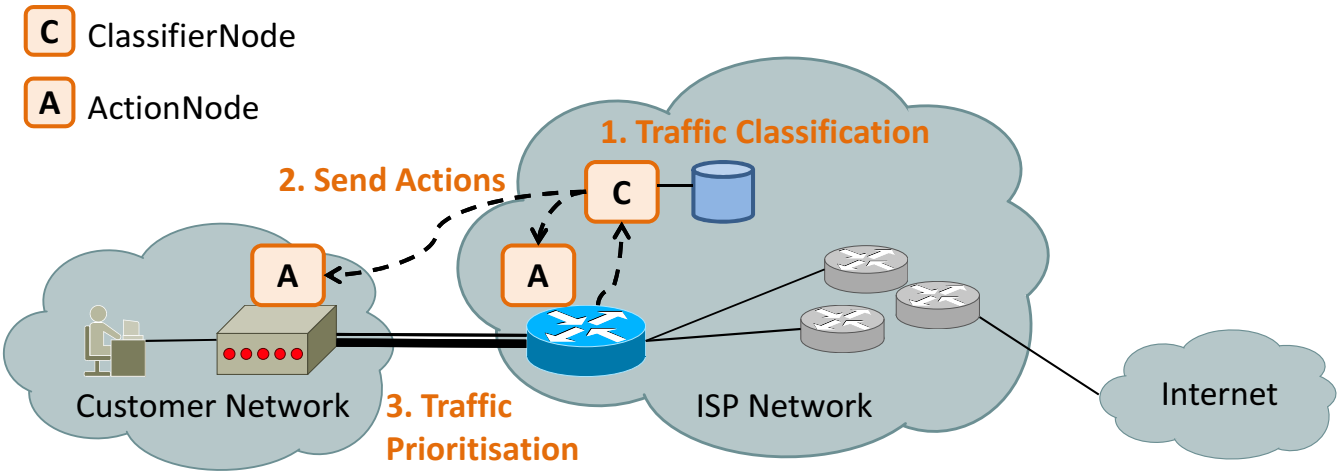
Use Case: Automated Traffic Prioritisation



Use Case: Automated Traffic Prioritisation



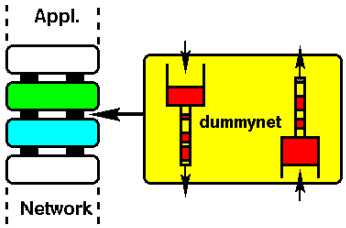
Use Case: Automated Traffic Prioritisation



SWINBURNE UNIVERSITY OF TECHNOLOGY
CAIA Seminar http://www.caia.swin.edu.au {szander, garmitage}@swin.edu.au October 14, 2010 16

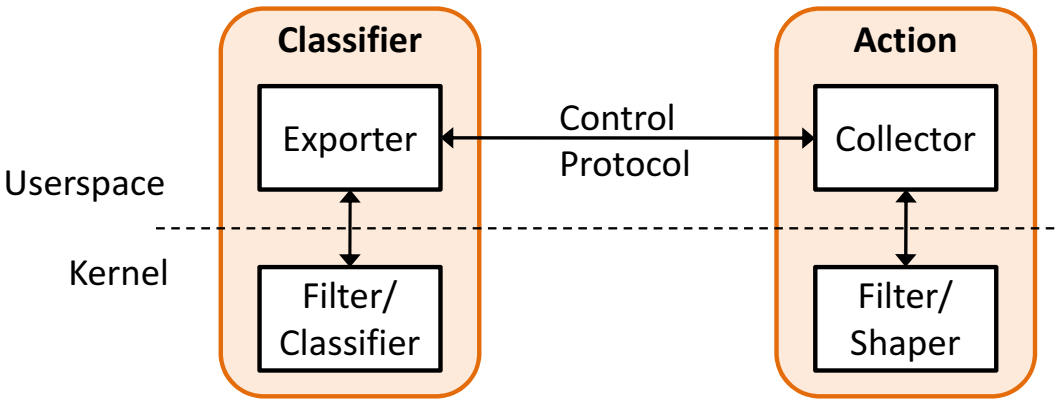
Choice of Firewall

- FreeBSD firewalls: IPFW, IPF, pf
- We choose IPFW/Dummynet
  - Dummynet shaper and interfaces to ALT-Q shaper
  - Runs on FreeBSD, Linux, Windows
  - FreeBSD's sponsored firewall
  - Used in previous CAIA projects
  - Code relatively well commented
  - Not the most features, but has all we need



Overall Architecture

- Classifier Nodes (CNs) classify flows based on features
- Action Nodes (ANs) treat flows based on rules received
- CNs control ANs via control protocol
- Extended rule language used to configure CNs, ANs



Extended rule set language

- Config commands for features, classifiers, rule exports

```
ipfw feature myplen config module plen window 10
ipfw mlclass myclass config algorithm nbayes model
game_vs_other
```

- Delete and show commands for features, classsifiers, exports

```
ipfw feature myplen delete
ipfw mlclass myclass delete
```

- Show command for features, classifiers, exports, flows

```
ipfw feature myplen show
ipfw flowtable show
```

- Feature matches

```
ipfw add allow tag 1 ip from any to any max.myplen>500
ipfw add allow tag 2 ip from any to any mean.myplen<=100
```


Extended rule set language

- New action for ML classifier, classifier matches

```
ipfw mlclass myclass ip from any to any use-feature-stats
fwd.min.myplen,...
ipfw count ip from any to any match-if-class myclass:0
ipfw count ip from any to any match-if-class myclass:1
```

- Classifier using IPFW tags

```
ipfw mlclass myclass ip from any to any use-feature-stats
fwd.min.myplen,...  class-tags 41,42
ipfw count ip from any to any tagged 41
ipfw count ip from any to any tagged 42
```

- New action for export

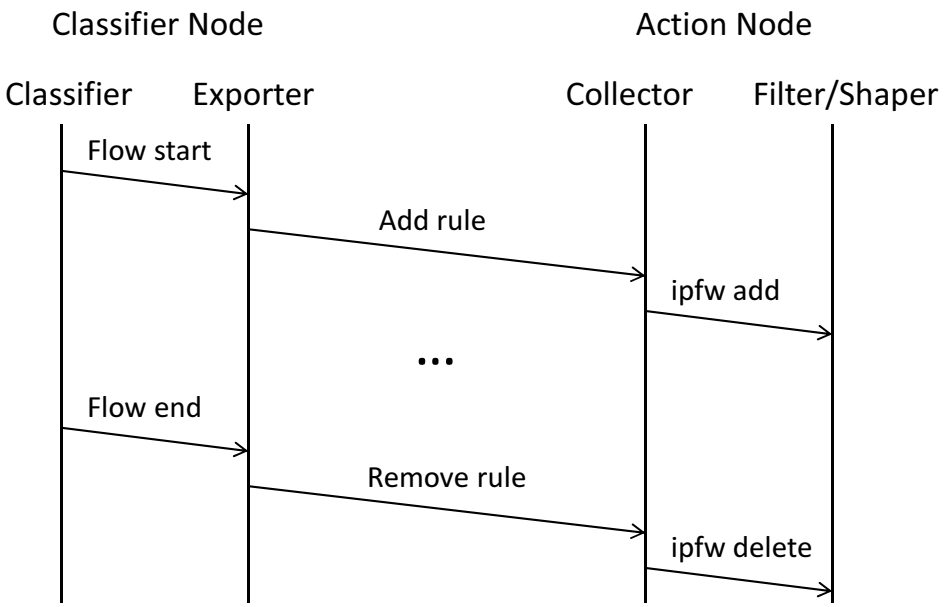
```
ipfw export myexp ip from any to any match-if-class
myclass:1
```

Control protocol

- Binary protocol between CNs and ANs
- Based on IETF IP Flow Information Export (IPFIX) protocol (RFC 3917, RFC 5051)
- Short fixed message header
- Templates define content of option and data sets
- Template management depends on transport
- Transport over SCTP (preferred), UDP, TCP

Control protocol

- Data sets contain rules for ANs
- Option sets convey extra information for ANs
- Add messages to send rules to ANs
- Remove messages / timeouts to delete rules from ANs

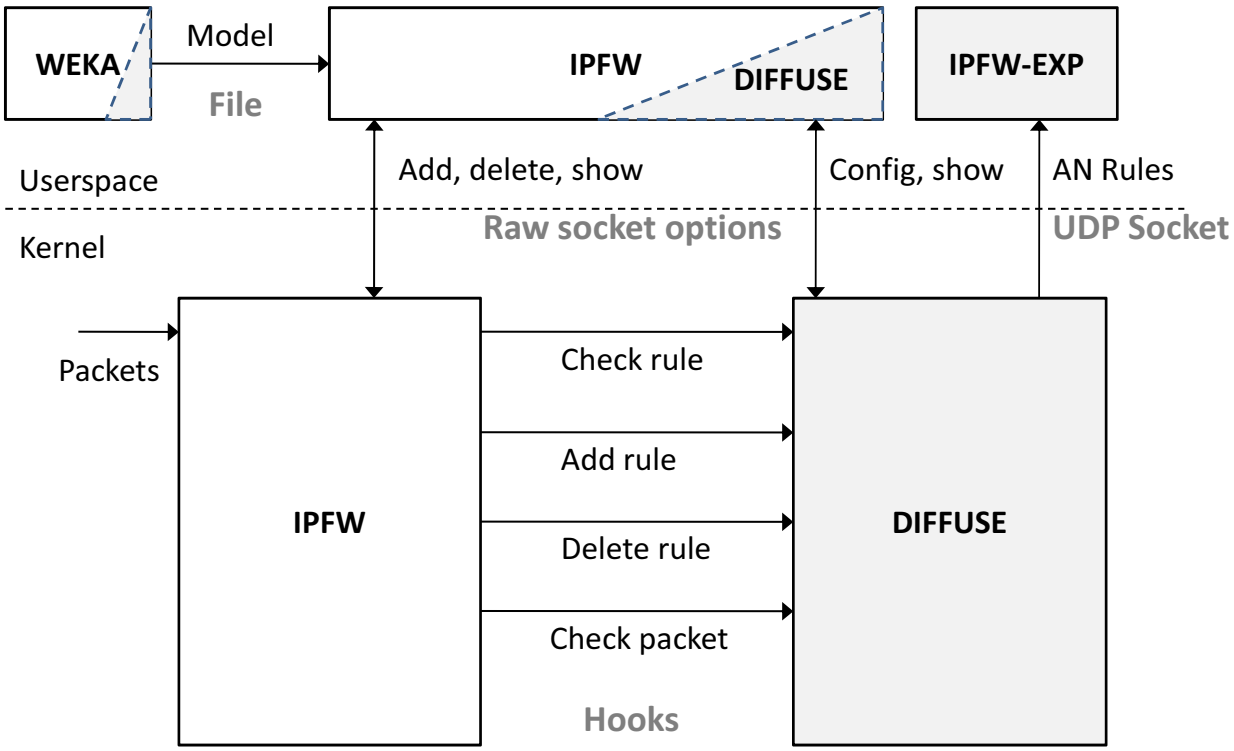


Control protocol

0	8	16	24	31
Version		Length		
Sequence number				
Timestamp				
Set ID = 1		Set length		
Template ID = 256		Flags		
ID = Src IP		ID = Dst IP		
...				
ID = Action		ID = Action Params		
Set ID = 256		Set length		
192.168.0.1		192.168.0.2		
...				
pipe		1		
192.168.0.3		192.168.0.2		
...				
pipe		2		

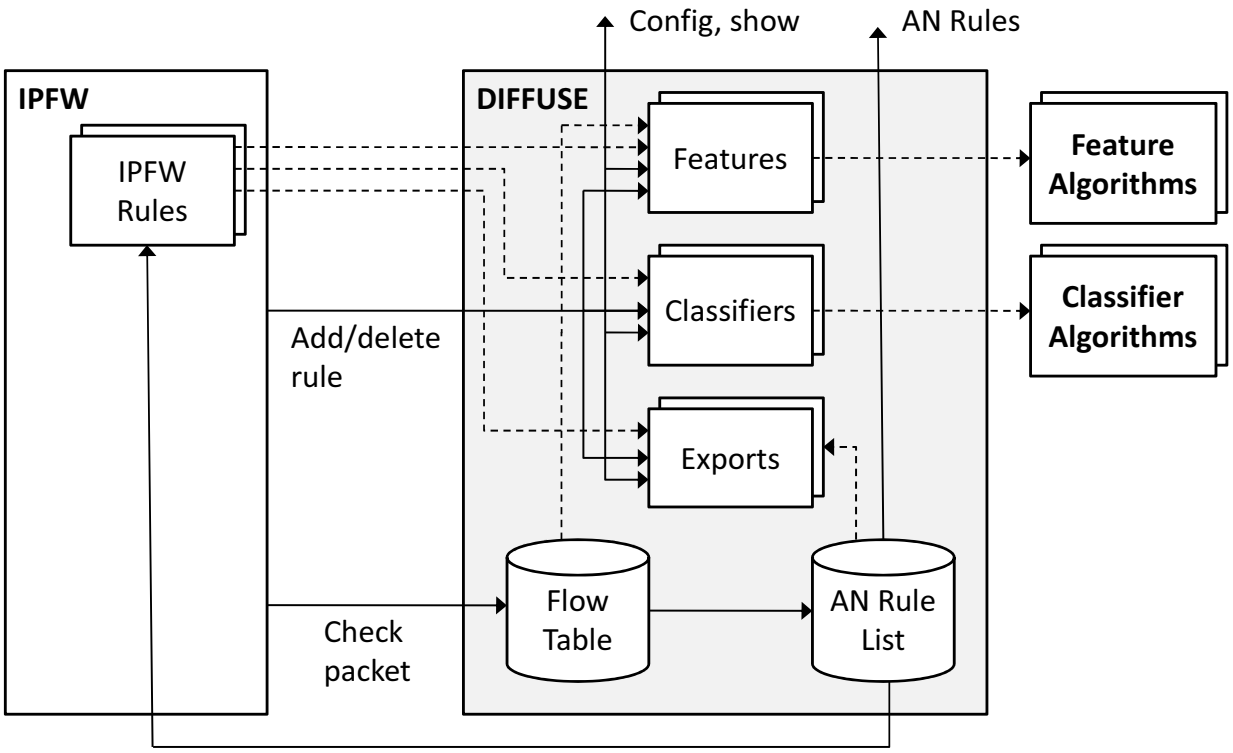
Implementation Classifier Node

- Main modules and interfaces



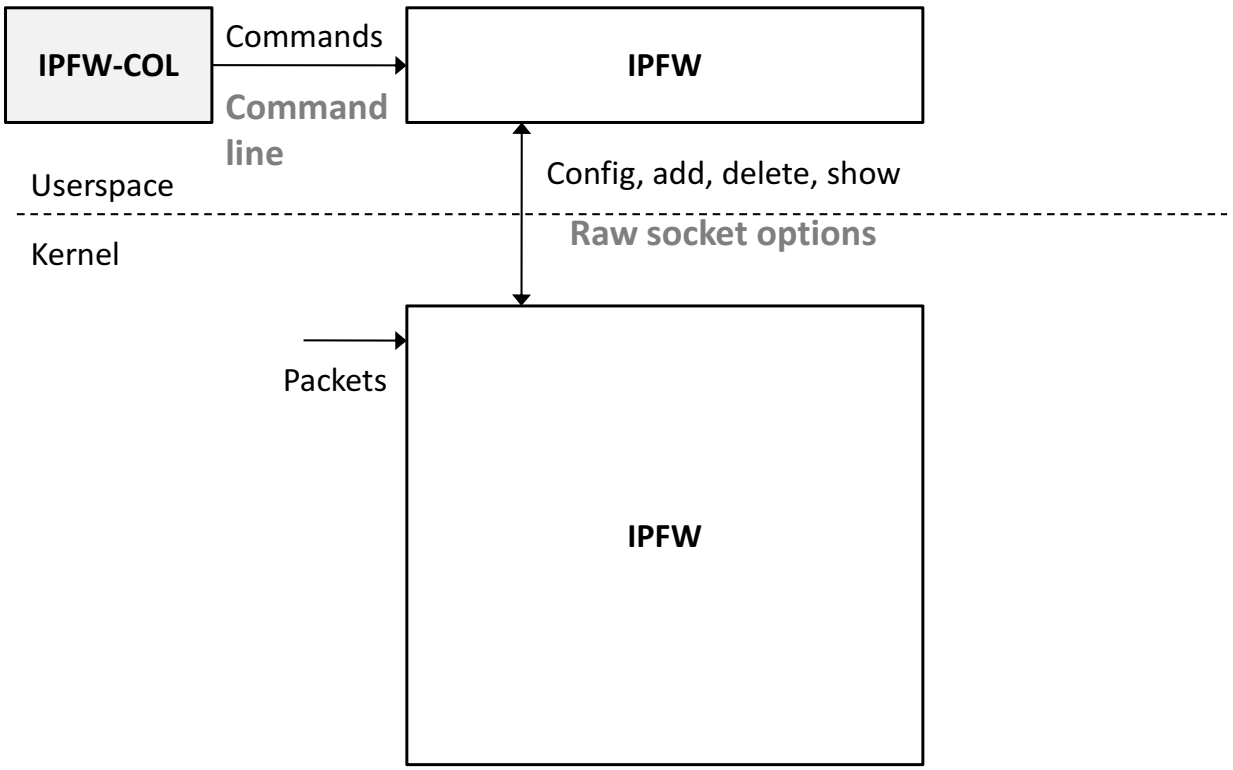
Implementation Classifier Node

- Main data structures and relations



Implementation Action Node

- Main modules and interfaces



Example output

- Example output, life demo next time

szander.caia.swin.edu.au - default* - SSH Secure Shell

File Edit View Window Help

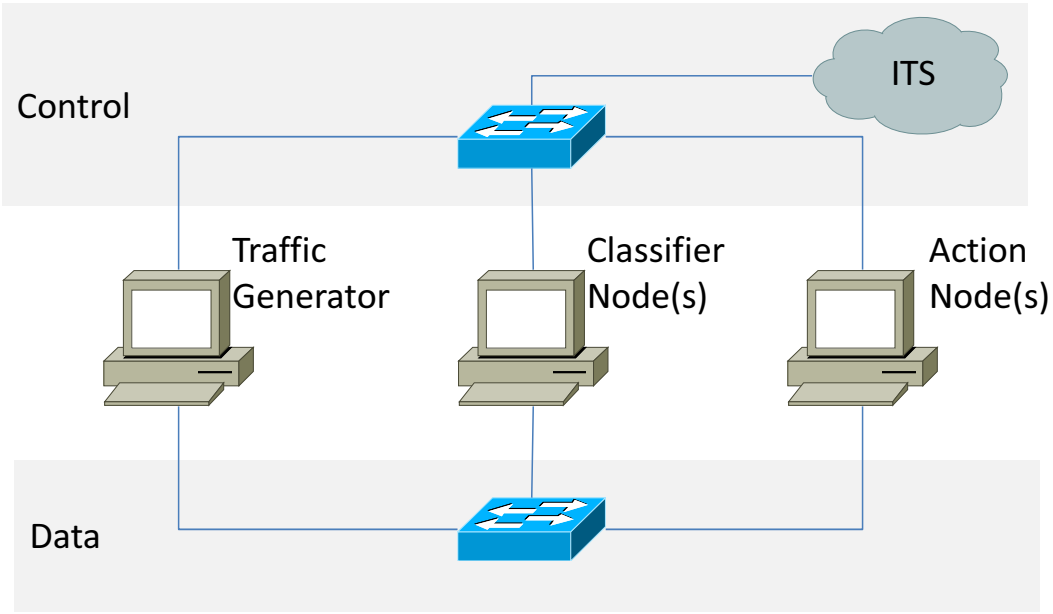
Quick Connect Profiles

```
[root@szander /home/szander/dev/diffuse_head/sbin/ipfw]# ipfw show
65535 2910 288800 allow ip from any to any
[root@szander /home/szander/dev/diffuse_head/sbin/ipfw]# ./test5.rules
00100  mlclass myclass ip from any to any features plen bidirectional use-feature-stats fwd.min.plen,fwd.
mean.plen,fwd.max.plen,bck.min.plen,bck.mean.plen,bck.max.plen
00200 count ip from any to any match-if-class myclass:0
00300 count ip from any to any match-if-class myclass:1
00400 export myexp ip from any to any match-if-class myclass:1
[root@szander /home/szander/dev/diffuse_head/sbin/ipfw]# ipfw show
00100   40   3488  mlclass myclass ip from any to any features plen bidirectional use-feature-stats fwd.m
in.plen,fwd.mean.plen,fwd.max.plen,bck.min.plen,bck.mean.plen,bck.max.plen
00200    0      0 count ip from any to any match-if-class myclass:0
00300   30   2432 count ip from any to any match-if-class myclass:1
00400   29   2392 export myexp ip from any to any match-if-class myclass:1
65535 2987 295352 allow ip from any to any
[root@szander /home/szander/dev/diffuse_head/sbin/ipfw]# ipfw show
00100   84   7344  mlclass myclass ip from any to any features plen bidirectional use-feature-stats fwd.m
in.plen,fwd.mean.plen,fwd.max.plen,bck.min.plen,bck.mean.plen,bck.max.plen
00200    0      0 count ip from any to any match-if-class myclass:0
00300   69   5832 count ip from any to any match-if-class myclass:1
00400   68   5792 export myexp ip from any to any match-if-class myclass:1
65535 3031 299208 allow ip from any to any
```

Connected to szander.caia.swin.edu.au SSH2 - aes128-cbc - hmac-md5 105x22

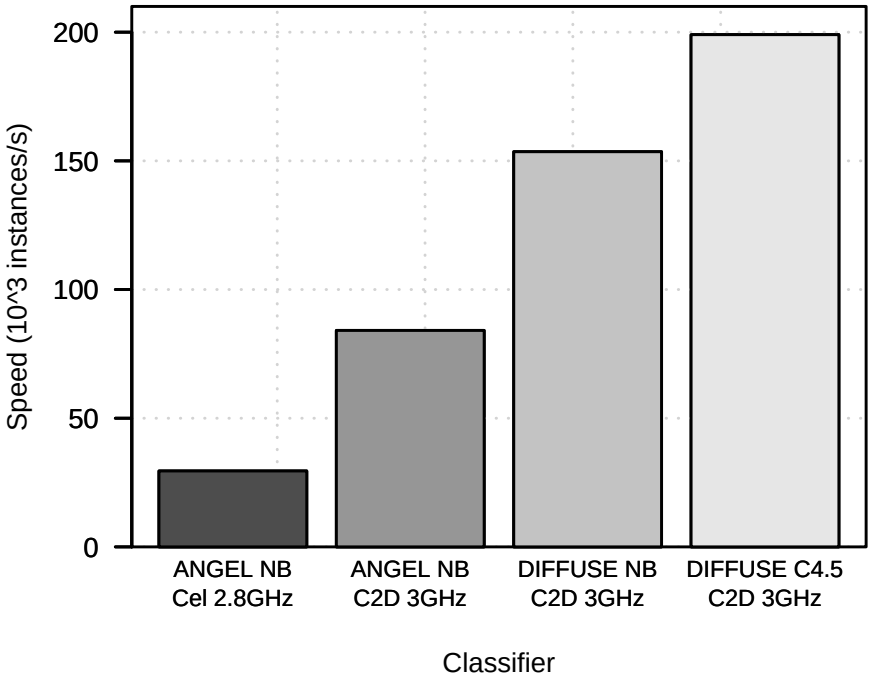
Testbed

- Evaluate functionality, performance



Preliminary Classification Speed Results

- Number of instances divided by mean execution time (as in ANGEL)
- Classify sub flow instances from ARFF file



Conclusions

- DIFFUSE is an IPFW extension that provides
  - Machine Learning based traffic classification
  - Spatial separation of flow classification and treatment
- Enables new application, such as automatic prioritisation of interactive traffic
- WEKA model output implemented
- Classifier implemented (some TODOs left)
  - Packet length, inter-arrival time features
  - Naïve Bayes and C4.5 classifiers
- ~6,500 LOC (IPFW/Dummynet is ~18,000 LOC)
- Initial version will be publicly available in Dec 2010

Future work

- Plenty to do in the remaining 7–8 months
- Implementation of exporter, collector, protocol
- Testing and documentation
- Evaluation of speed, accuracy
- Support passive measurements (promiscuous packets)
- Research stability of classifiers and retraining mechanisms
- Integration and evaluation of Skype/BitTorrent features (Rozanna, Philip, Jason)