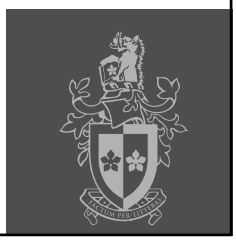
SWINBURNE

CENTRE FOR ADVANCED INTERNET ARCHITECTURES

Machine Learning Algorithms for Automated Network Application Identification – Work in progress

Nigel Williams, Sebastian Zander

{niwilliams,szander}@swin.edu.au



Presentation Outline

- Motivation
- Our Proposal
- Experimental Setup
- Results
- Conclusions and Further Work

Motivation

- Key areas in IP network engineering, management and surveillance that greatly benefit from classifying flows according to their responsible applications
  - Application-based traffic engineering, monitoring
  - Adaptive, network-based QoS mapping
  - Dynamic application-based access control
  - Lawful interception
  - Intrusion detection
- Current solutions only moderately accurate and/or difficult to implement

Current Solutions…

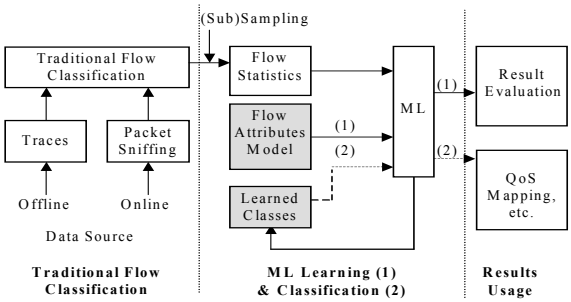
- Port-based identification
  - Well-known and registered ports (IANA)
  - Known default ports (eg http://www.portsdb.org)
- Stateful reconstruction of session and application information
  - Inspect packet payload for application specific protocol data
- Signature-based identification
  - Payload pattern recognition

… and their Shortfalls

- Port-based
  - Applications do not always use the specified port, or use unknown ports
  - Running multiple servers on same machine
  - Users using different ports to bypass port-based filters
  - Ports allocated dynamically (e.g. passive FTP, streaming)
- Stateful Reconstruction
  - Resource intensive, must have protocol knowledge of each application to be classified
  - Does not work when payload encrypted
  - Privacy issues
- Signature-based approach
  - Less accurate than stateful reconstruction
  - Still requires knowledge of application protocols
  - Still ineffective when payload encrypted

Our Proposal

- Use Machine Learning (ML) techniques to classify flows based on flow attributes
  - Use supervised learning algorithms
    - Algorithm is trained using examples of each traffic class
  - Use flow attributes that are protocol independent
    - Packet level: e.g. packet length
    - Flow level: e.g. inter-arrival times, duration



Approach

- Which attributes are most useful? → Feature Selection
  - Identify 'important' features
  - Minimise number of features
  - Compare against full set of features
- What kind of algorithm? → Evaluate algorithms
  - Test algorithms using different subsets of features

Feature Selection

- Feature set describes data instance (in our case a network flow)
  - Quality of features greatly influence the effectiveness of ML algorithm
  - Decreasing the feature space can improve accuracy and provide faster classification times
  - Want to avoid redundant or irrelevant features because some algorithms don't like them
- How do we identify good features?
  - Use feature selection algorithms
  - Two feature selection models – Wrapper and Filter

Filter Model

- Use some goodness measure to compare features or subsets of features
  - Ranking and Subset Search algorithms
- We use two subset search algorithms
  - Consistency-Based Subset Search (CON)
  - Correlation-Based Feature Selection (CFS)
  - Generate subsets of full feature set, generate score for each subset
  - Selected subsets are not biased towards any ML algorithm

Filter Model contd.

- Consistency-based
  - Identifies smallest subset that will identify a class as consistently as the full feature set
  - Evaluates the consistency between features describing a class within each subset
- Correlation-based
  - Subset evaluation heuristic examines usefulness of features for predicting class, as well as inter-correlation between features
  - Want subsets containing attributes that are highly correlated with the class and have low inter-correlation with each other

Wrapper Model

- Use the performance of a pre-determined algorithm to determine best subset
- Will identify a subset 'tailored' to the chosen algorithm
- Expect better results than with filter model selection, but comes at a processing cost
  - Must repeatedly execute ML algorithm
  - One evaluation took a month to complete

Generating Subsets

- Need to provide candidate subsets for the feature selection algorithms to evaluate
- Use a search algorithm
  - Best First (forward, backward)
  - Exhaustive
  - Greedy (forward, backward)
  - Genetic

ML Algorithms

- Mostly supervised algorithms (inductive learning)
  - Forms a model based on training data and uses model to classify unseen new data
  - Models can be represented by trees structures, probability tables…
  - We use ML algorithms with a range of model representations
- One unsupervised algorithm (clustering)
  - No training, clusters data into different classes based on similarities and differences of each instance
  - For comparison purposes

ML Algorithms

- C4.5 Decision Tree
- Naive Bayes (NB)
- Bayesian Networks (BN)
- Nearest Neighbour (NN)
- Naive Bayes Tree (NBTree)
- Multilayer Perceptron Network (MLP)
- Sequential Minimal Optimisation (SMO)
- Expectation Maximisation (EM, unsupervised)
- Also use Adaptive Boosting for C4.5 and Naive Bayes

C4.5

- Creates a decision model based on a tree structure of nodes, branches and leaf nodes
- Nodes represent features, branches values connecting the features and leaf nodes represent class
- Models are easy to interpret
- Performs feature selection as part of training process

Naive Bayes

- Based on the Bayesian Theorem
- Uses conditional probability and prior probability
- Assumes independence of features
- Uses continuous probability distributions to model attributes
- Use Bayes with Discretisation and with Kernel Density Estimation

Bayesian Networks

- A combination of a graph of nodes and links, and a set of probability tables for each node
- Nodes represent features or class, links the relationship between them.
- Uses a search algorithm to create node structure
  - K2 Hill-climbing
- Probability estimation algorithm
  - Simple Estimator

Nearest Neighbour

- Predictive ‘lazy’ learner – does not predict based on model of training data, training data is the model
- Predicts class by finding the most similar training instance stored in the model (based on a distance metric)

Naive Bayes Tree

- A hybrid decision tree and Naive Bayes classifier.
- A decision tree of links and nodes with Bayesian classifier on the leaf nodes

Multilayer Perceptron Network

- Neural Network based on back propagation
- 'Black box' model
- Backpropagation network with sigmoid threshold functions
- No. input nodes = No. of attributes and No. output nodes = number of classes (six in our case)
- Complex setup

Sequential Minimal Optimisation

- An extension of Support Vector Machines
- Maps data into high-dimensional space to achieve a linear separation of class
- Actually a binary classifier
  - Train multiple classifiers for multi-class scenario
- Complex setup

Expectation Maximisation (EM)

- Unsupervised ‘clustering’ algorithm
- Iterative optimisation method
- We use two configurations of EM
  - Maximum number of clusters equal to number of classes
  - No limit to number of clusters (EM allowed to identify ‘best’ number of clusters)

Experimental Setup

- Weka Software
  - Waikato Environment for Knowledge Analysis
  - Open source
  - Java implementations of many ML algorithms
- NetMate
  - Extensible network measurement meter
  - Open Source
  - Calculates flow attributes
- Trace Files
- Traffic Classes
- Features

SWIN BUR NE — CENTRE FOR ADVANCED INTERNET ARCHITECTURES

Data Traces

- Three publicly available NLANR traces
  - Leipzig, Auckland and Wellington (NZIX)
  - Traces captured in different years
  - Use flow data from four 24-hour periods from these traces
  - These periods are further sub-sampled
- Public trace files are anonymised
  - Cannot verify packet payload (above transport layer)

SWIN BUR NE — CENTRE FOR ADVANCED INTERNET ARCHITECTURES

Data Traces

Trace	# of flows
Auckland-vi-20010611	6,000
Auckland-vi-20010612	6,000
Leipzig-ii-20030221	5,254
NZIX-ii-20000706	4,743
Total	21,997

Traffic Classes

- Need to provide training examples of different applications for the classifier
  - Obtain training examples from the (anonymised) data traces
- Must define classes according to IANA registered ports
  - Can we rely on port numbers?
  - Choose applications with high likelihood of correct port usage
  - Worst case is a reduction in classification accuracy…

Traffic Classes

- Our classes make up about 75% of Auckland and NZIX flows and bytes
- An excessive amount of peer-to-peer activity means our traffic classes represent only a fraction of the Leipzig flows and bytes

Class	Description
20	FTP-Data
23	Telnet
25	SMTP
53	DNS
80	HTTP
27015	Half-Life

	Percentage of Flows / Bytes [%]			
Port	**Auck-11**	**Auck-12**	**Leipzig**	**NZIX**
20	0.2 / 2.6	0.2 / 1.5	0.1 / 0.6	0.5 / 4.3
23	0.1 / 0.1	1.8 / 0.1	0.1 / 0.1	0.1 / 0.1
25	2.0 / 6.7	2.6 / 6.1	0.4 / 0.4	2.8 / 18.8
53	3.7 / 0.6	3.9 / 0.5	2.0 / 0.1	16.6 / 2.4
80	67.0 / 66.7	64.5 / 61.2	11.3 / 20.3	52.5 / 49.8
27015	0.7 / 0.1	1.3 / 0.1	0.7 / 0.1	0.1 / 0.6
Sum	73.7 / 76.8	74.3 / 69.5	14.6 / 21.6	72.6 / 76

Features

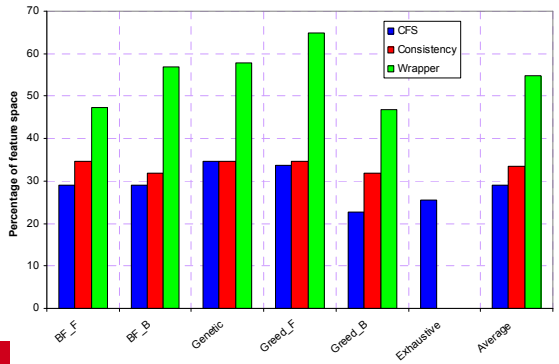
- Use NetMate to process packet traces and compute features
- Flows are bi-directional (forward and backward directions)
- Flow Timeout fixed at 60s
  - For initial trials
- Attribute set based on
  - Packet Inter-arrival time
  - Packet Length
  - Flow Size (bytes, packets)
  - Duration
  - Protocol
- Compute minimum, mean, maximum standard deviation…
- Attributes are bi-directional (except duration)
- No protocol specific attributes are used

Evaluation Metrics

- Overall accuracy
- Rate Metrics
  - Precision (Class members classified correctly over total instances classified as class members)
  - Recall (Class members classified correctly over total number of class instances)
  - False Positive
  - False Negative
- Training, classification time
- Use 10-fold cross validation
  - Divide data into 10 subsets, perform classification 10 times, each time with 9 training and 1 testing subsets, results are average across all trials

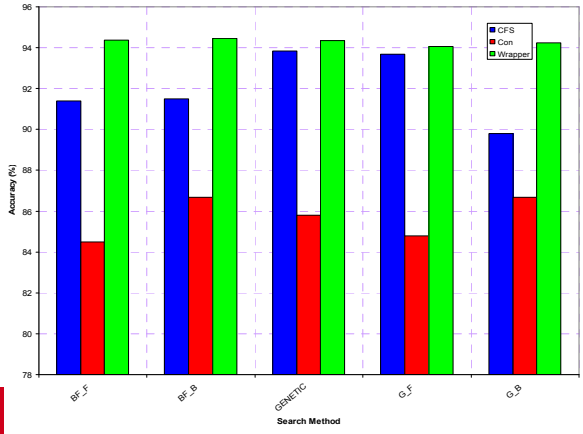
Results – Feature Selection

- Feature space reduction by subset evaluation method and search method
- Percentage of original feature space (average across algorithms)
  - Wrapper reduction varies according to ML algorithm



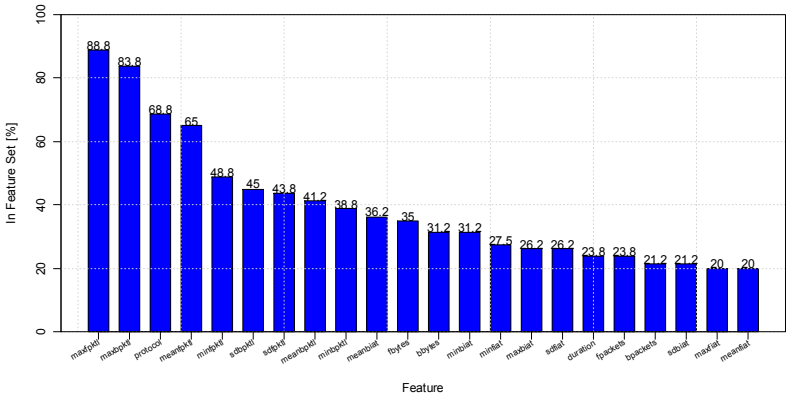
Results – Feature Selection

- Mean accuracy according to search method
  - Little difference in achieved accuracy with same subset evaluator



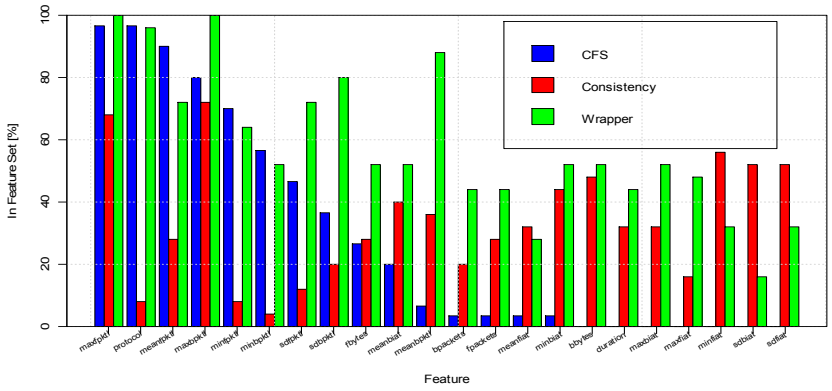
Results – Feature Selection

- Percentage of subsets in which feature was selected



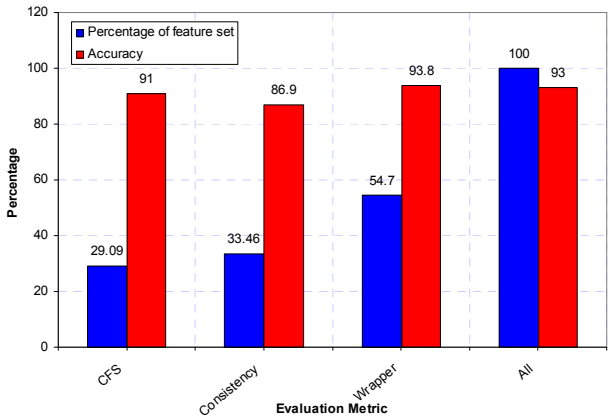
Results – Feature Selection

- Percentage of subsets in which feature selected, by evaluation method



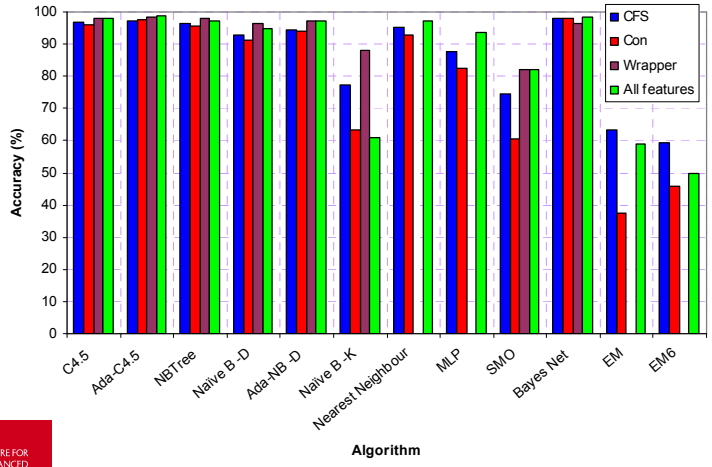
Results – Feature Selection

- Comparing subset evaluators to full feature set



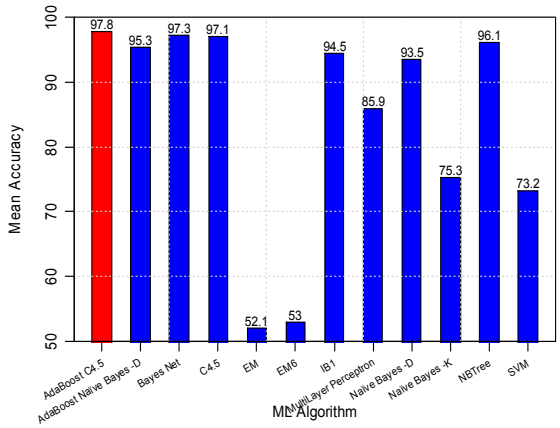
Results – ML Algorithms

- Average accuracy for different feature sets by subset evaluation method



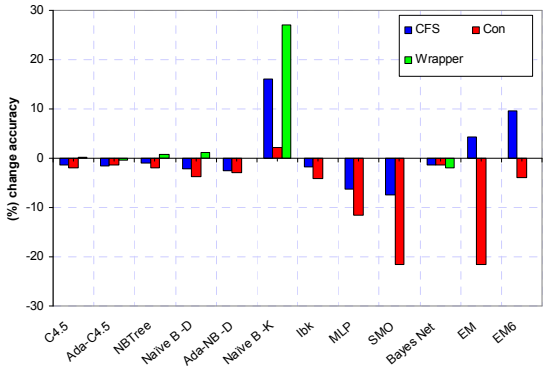
Results – ML Algorithms

- Mean accuracy of algorithms
- EM performs poorly
- Decision trees and Bayesian classifiers provide best results



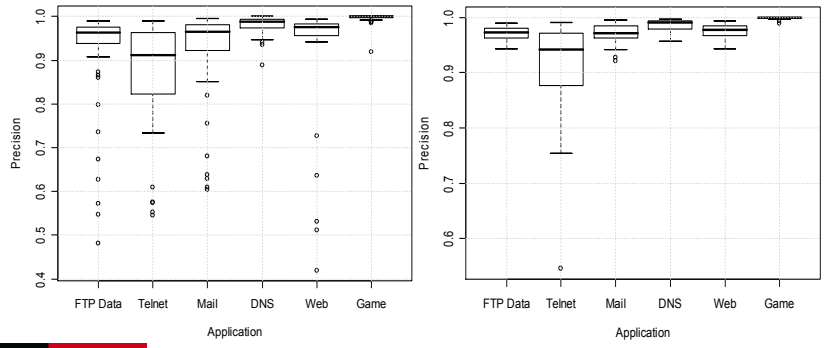
Results – ML Algorithms

- Effect of feature selection on algorithm accuracy
- MLP, SMO and EM prefer larger feature sets
- Naive Bayes with Kernel Density Estimation benefits greatly from CFS and Wrapper feature selection



Results – ML Algorithms

- Mean Precision rate per application class
  - Telnet most difficult to classify (not enough training flows?)
  - Half-Life traffic most unique
  - Web performance better than expected



All-algorithms Better-algorithms

Results – So far…

- CFS provides most aggressive reduction in feature space and better accuracy than Consistency
- Wrapper method subsets on average produce more accurate classification than the full feature set (while reducing the feature down to almost 50%!)
- Little difference in achieved accuracies between search methods (especially for wrapper)
- C4.5, Naive Bayes with Discretisation, NBTree and Bayesian Networks show strongest performance
  - More criteria to required to discriminate the algorithms

Additional Criteria

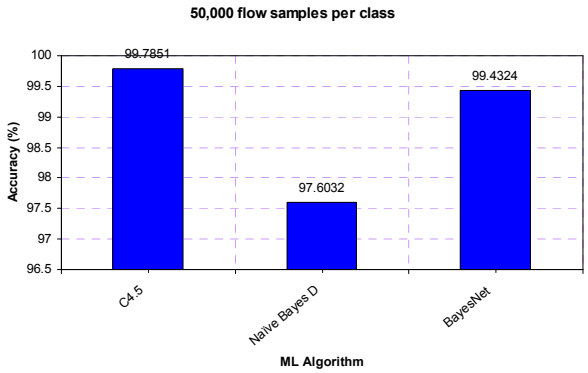
Algorithm	C4.5	AdaBoost C4.5	Naive Bayes - D	NBTree	Bayes Net	AdaBoost Naive Bayes –D
Average Accuracy for selection methods [%] (combined trace)	96.88	97.89	92.45	96.26	96.29	94.28
Build Time [s]	26.61	319.11	10.29	1318.56	7.78	173.04
Classifications per second	52,631	3,133	22,222	5,988	17,857	1,412
No. of features for best accuracy	All	All	8	8	All	15
Setup Complexity	Moderate	Moderate	Low	Moderate	Moderate-High	Low
Model Interpretability	Very High	Very High	Moderate	High	High	Moderate

New Datasets

- Revised flow definitions
  - UDP flows:
    - at least one packet in each direction
    - 600s timeout
  - TCP flows:
    - at least one packet in each direction and at least 1 byte data transferred
    - exported on RST or FIN/ACK or after 600s timeout

New Datasets

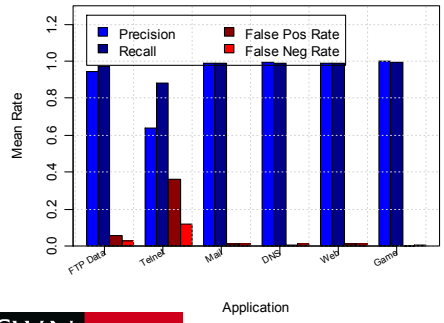
- Combined dataset, including new flows from Waikato trace
- Up to 50,000 flows of each traffic class
- 828,734 instances in total
- Use all features



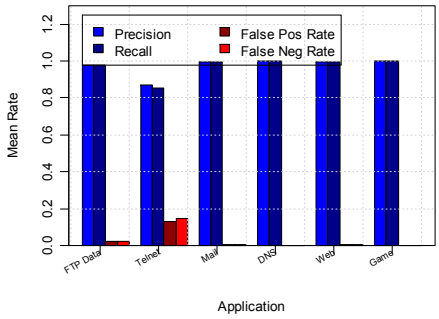
New Datasets

- Mean metric rates per application class
  - Again telnet most incorrectly classified (mainly by Naive Bayes, most likely due to lack of training instances -> low prior probability)

With Naive Bayes

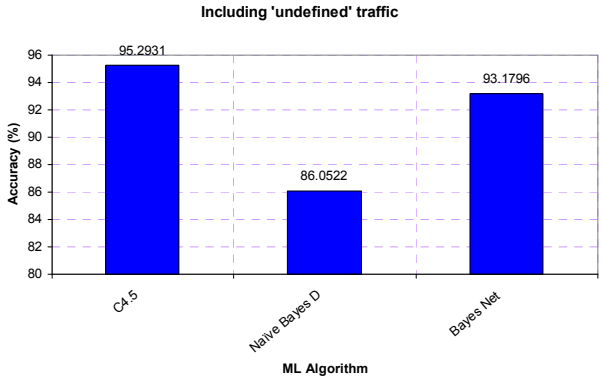


C4.5/Bayes Net only



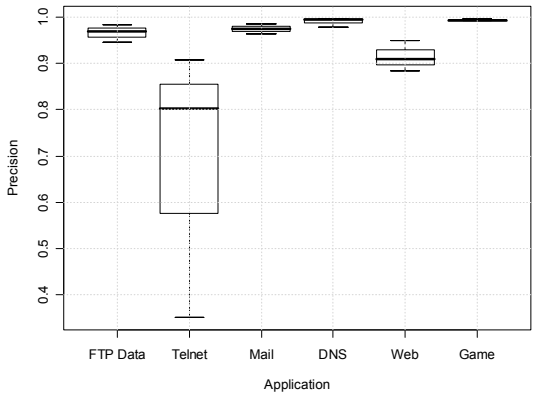
New Datasets

- Trained with new class ‘other’ containing samples of all other flows in the trace files
- Up to 10,000 samples for each flow
- 251,717 instances in total



New Datasets

- Mean Precision rate per application class
  - http precision decreased, possibly identifying remote admin tools, www tunnel Trojans..



Conclusions & Future Work

- Great potential in identifying IP network traffic using protocol-independent flow derived statistics
  - Able to identify several traffic classes with very high accuracy
  - Our current feature set showed good discriminating power
  - Identified a number of suitable algorithms (C4.5, Naive Bayes..)
- Further avenues of interest..
  - Explore new features (idle/active times..)
  - Effect of flow sampling, flow timeout, [packet sampling]
  - How long until application can be reliably detected?
  - How stable are classifications over the flow lifetime?
  - Identify possible misclassifications
  - Larger training datasets, more traffic classes
  - Verified training data (generate in test-bed)
  - Detect applications in encrypted traffic

- More info at http://caia.swin.edu.au/urp/dstc/
- Thanks for listening
- Any Questions?

SWIN BUR NE

CENTRE FOR ADVANCED INTERNET ARCHITECTURES