Anomaly based techniques for Intrusion Detection Systems

Suseela Sarasamma,

Julie Huff

Northrop Grumman Mission Systems

NORTHROP GRUMMAN

Organization

1. **Anomaly detection**
    1. Definition
    2. General techniques
        1. Statistical
        2. Neural network
        3. Machine learning
2. **Intrusion detection**
        1. Categories
        2. Algorithmic techniques
3. **Anomaly detection techniques in IDS**
        1. A novel anomaly detection technique using Kohonen network
        2. Conclusions from the tests conducted on multi-level K-map

NORTHROP GRUMMAN

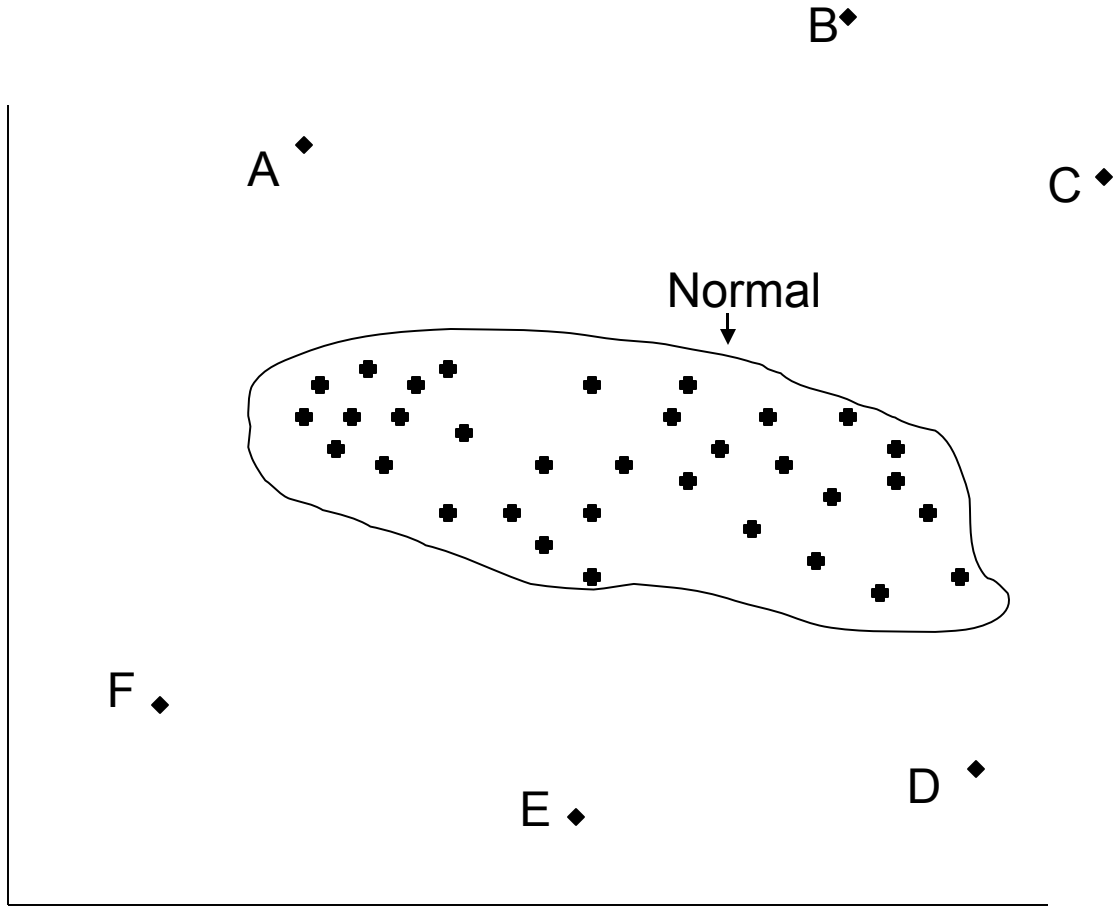
Anomaly Detection

- **Outlier or Anomaly: (Barnett and Lewis, 1994)**
  - An observation that appears to deviate markedly from the other members of the sample in which it occurs.
  - An observation that appears to be inconsistent with the remainder of that set of data.

- **Anomaly detection approaches are also described as**
  - Outlier detection
  - Novelty detection
  - Noise detection
  - Deviation detection
  - Exception mining

NORTHROP GRUMMAN

A graphical example



Points A, B, C, D, E, F represent outliers

Anomaly Detection Methodologies

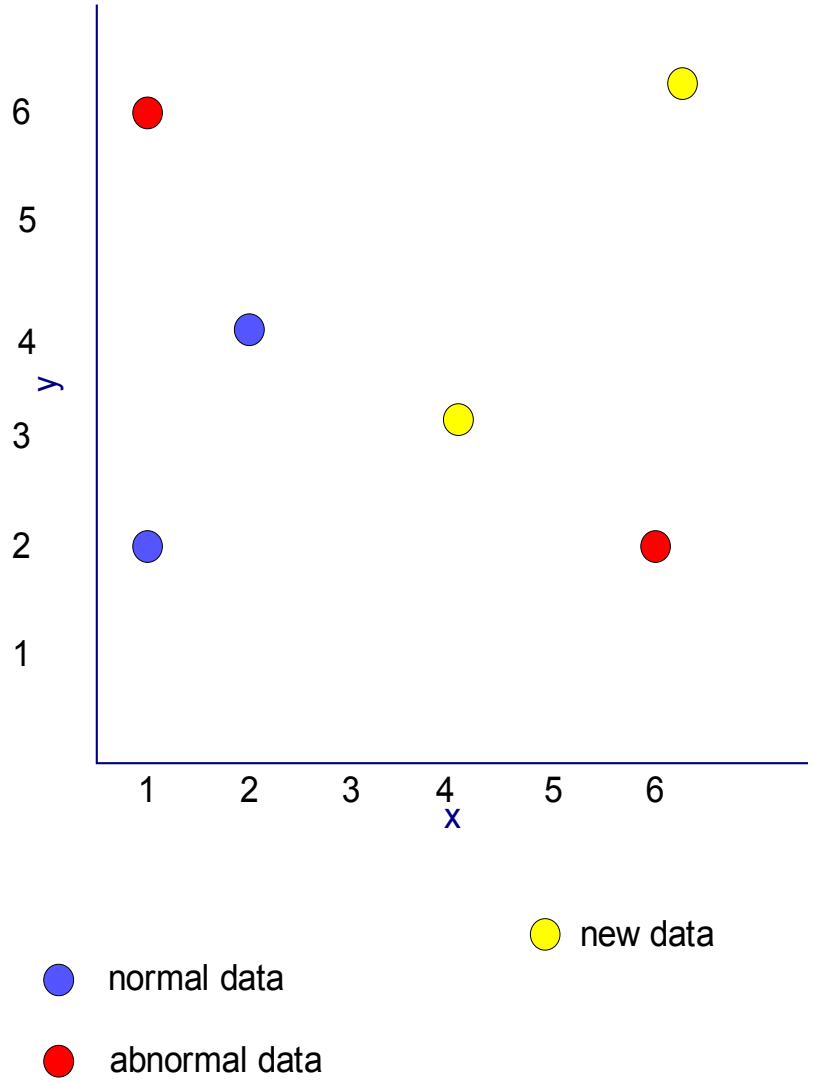
- **Statistics-based**
  - Proximity-based
  - Parametric
  - Non-parametric
  - Semi-parametric
- **Neural networks-based**
  - Supervised
  - Unsupervised
- **Machine learning**

NORTHROP GRUMMAN

Statistical techniques

- **Oldest technique**
- **Susceptible to the number of exemplars in the data set**
- **Generally suited for**
  - quantitative real-valued data or
  - quantitative ordinal data distributions
- **Proximity-based techniques:**
  - No prior assumptions about the data distribution model
  - Exponential computational complexity due to the need to calculate the distances of all data records.
  - Types of proximity-based algorithms
    - K-Nearest Neighbor (k-NN) algorithms
      - The learning phase groups the training vector space to a priori known classes
      - Calculates the k nearest neighbors of an unknown sample using a distance metric such as Euclidean distance or Mahalanobis distance
      - Assigns the unknown sample to the most common class among the k nearest neighbors

K nearest neighbor – graphical example



K-means clustering algorithm

- **Training algorithm**
  - Input: samples: n; # partitions : k; k < n
  - Output: k clusters that minimize sum of squared error
  - Algorithm:

    Arbitrarily choose k samples as initial cluster centers

    Repeat

    Assign each object to the cluster that the object is most similar, based on mean value of objects in cluster.

    Update cluster means

    Until no further change
- **Detecting anomaly:**
  - Compare unknown sample to the k prototype vectors
  - Choose the closest
  - Computationally more efficient compared to k-NN
- **Clustering is sensitive to noise and outlying values**

K-medoids algorithm

- **K-medoids algorithm**
  - Each cluster center is represented by a centrally located (the medoid) point rather than a prototype point.
  - Considered less susceptible to local minima compared to k-means clustering.
  - Independent of data-order compared to standard K-means clustering.
  - Provides better class separation than k-means
  - Can be computationally costlier compared to K-means
    - Up to $O(n^2)$ per iteration compared to $O(n)$ for K-means.

Parametric methods

- **Data points are modeled using pre-selected stochastic distribution model. Points are deemed outliers based on their relationship with the model.**
- **Fits boundaries around specific percentage of the data irrespective of the sparseness of the outlying region.**
- **If the user has a priori knowledge that the data set fits such a model, then highly accurate results are obtained.**
- **Rely on good spread for the data. Otherwise many normal points will be omitted from the bound of normality.**
- **Techniques suited for lower-dimensional data**
  - Minimum Volume Ellipsoid estimation (Rousseeuw and Leroy)
  - Convex Peeling
- **Techniques for higher dimensional data**
  - Principal component analysis

NORTHROP GRUMMAN

Non-parametric methods

- No assumptions about the underlying data distribution
- More suited for smaller sample size
- Number and nature of parameters is flexible
- Confidence bounds can be much wider than that for the parametric counterparts.
- Predictions outside the range of observations are not possible.
- Example : Chi Square estimate

NORTHROP GRUMMAN

Semi-parametric methods

- Combine the speed and complexity growth advantage of parametric methods with the model flexibility of non-parametric methods.
- Employ techniques that use kernels whose width are autonomously determined by the spread of the data.
- The density distribution of the input space is estimated and outliers are considered as points lying in low-density areas of the input space.

NORTHROP GRUMMAN

Neural Network-based methods

- **Generally non-parametric and model-based**
- **Generalizes well to unseen patterns**
- **Capable of learning complex class boundaries**
- **Needs training that iterates over the training set until the network settles**
- **Once trained, the neural network acts as a classifier.**
- **Higher dimensionality of data adds to computational cost, but not so much as in statistical techniques**
- **Have an input layer, an output layer and one or more hidden layers**
- **Two basic groups:**
  - Supervised neural networks
    - The learning process requires the data to be labeled and classified
    - Examples:
      - Multi-layer Perceptron, a feed-forward neural network
      - Hopfield network, an auto-associative neural network
      - Radial basis function networks
  - Unsupervised neural networks
    - The learning process does not require pre-classified data
    - The nodes in the competitive (hidden) layer compete to represent portions of the data
    - Training iterations cluster the input vectors to model the underlying data distribution
    - Examples:
      - Self organizing maps (Kohonen maps)
      - Adaptive Resonance Theory (ART) networks

Machine learning approaches

- **To process categorical data that has no implicit ordering.**
- **Do not need prior knowledge of data.**
- **Common techniques applied:**
  - Decision trees
    - Simple class boundaries
    - Works well with noisy data
    - Accommodates large data sets and large dimensions
    - Susceptible to over fitting
  - Rule-based techniques
    - Tests series of conditions to produce conclusions
    - More flexible and incremental than decision trees
      - New rules can be added
      - Existing rules can be amended

      without disturbing existing rules
    - May be a classifier that learns classification rules from both normal and abnormal data, or a recognizer trained on normal data alone.

Intrusion Detection Systems

- **From an architectural perspective, an Intrusion Detection System (*IDS*) can be one of three basic types :**
  - network-based IDS
  - host-based IDS
  - hybrid IDS.
- **Network-based IDS monitor the IP packets, usually packet headers flowing over the network.**
- **Host-based IDS use the audit data of host machines.**
- **Hybrid IDS apply both the above methods to capture intrusions from outside as well as from within.**

NORTHROP GRUMMAN

Intrusion Categories

- **Probe events**
  - Scanning operations to glean info about active hosts and possible vulnerabilities in the network
  - examples: ipsweep, portsweep, nmap, satan, saint
- **Denial of service type events**
  - Deplete the network resources such as bandwidth, connectivity by issuing high volume of traffic or high volume of connection requests
  - degrades the performance and sometimes crashes the system
  - Acts such as broadcasting multiple echo requests (may be from a spoofed address) to a target network. When each of the hosts in the victim network responds with an echo reply, the performance is severely degraded.
  - Deny access to legitimate users
  - Malicious fragmentation (teardrop, ping of death)
  - Sending a SYN packet with the victim's IP address in both source and destination (land)
  - SYN flooding with half-open SYN connections
  - examples:
    - Smurf, teardrop, pod, back, land, apache2, udpstorm, Mailbomb, …

NORTHROP GRUMMAN

Intrusion categories – contd.

- **User to Root type events**
  - Attacker has access to a normal user account on the system.
  - Exploits some vulnerability to gain root privilege
  - Imagine the rest …
  - Examples:
    - perl, ps, xterm, loadmodule, eject, buffer_overflow, rootkit, …
- **Remote to local type events**
  - Attacker does not have an account
  - Exploits some vulnerability in the network to gain local access
  - Sends packets to the compromised machine over the network
  - Examples:
    - dictionary, ftp_write, guess_passwd, imap, named, sendmail, spy, xlock, xsnoop, …

NORTHROP GRUMMAN

Commonly used techniques for IDS

- **Algorithmically, there are two different approaches commonly used in detecting intrusions.**
  - Misuse detection, a rule-based approach that uses stored signatures of known intrusion instances to detect an attack.
  - Anomaly detection, a data-driven approach that uses data-mining techniques.

NORTHROP GRUMMAN

Misuse detection

- A rule-based approach that uses stored signatures of known intrusion instances to detect an attack.
- Highly successful in detecting occurrences of previously known attacks.
- Fails to detect new attack types and variants of known attacks whose signatures are not stored.
- When new attacks occur, the signature database has to be manually modified for future use.
- Example: The SNORT System

NORTHROP GRUMMAN

Anomaly Detection Approaches in IDS

- **Profile of perceived normal behavior is established**
- **Deviants from the normal profile are considered anomalies or potential attacks.**
- **Normal operations that exhibit behavior adherent to unseen mode of operation may be detected as anomalies.**
  - Such cases of false detection are termed false positives.
- **No exact templates to match an unknown event**
- **The hall-mark of a good anomaly-based approach:**
  - high detection rate at a low false positive rate.
- **Merit of an anomaly detection scheme:**
  - absence of an enormous signature database.

NORTHROP GRUMMAN

Modeling anomaly detection problem in NIDS

- Capture network packet headers using tools such as tcpdump, ethereal.
- Process dump data into connection records
  - Tcptrace , open source
- A connection record encapsulates much of the basic TCP/IP characteristics of all the IP traffic during the lifetime of a connection.
- Map attributes in connection record to numerical values.
- A connection record with n attributes will thus map to an n-dimensional vector.
- Apply one or more of the standard anomaly detection techniques on the above vector space.
- Identify the records corresponding to the anomaly (outlier) vectors.

NORTHROP GRUMMAN

Some features available in a connection record

- **Some Basic features:**
  - Duration of connection
  - Service used
  - Source host
  - Destination host
  - Source port
  - Destination port
  - #bytes transmitted from source
  - #bytes transmitted from destination
  - #packets from source to destination
  - #packets from destination to source
  - Protocol used
  - Flag
  - Number of wrong fragments
  - Number of urgent packets

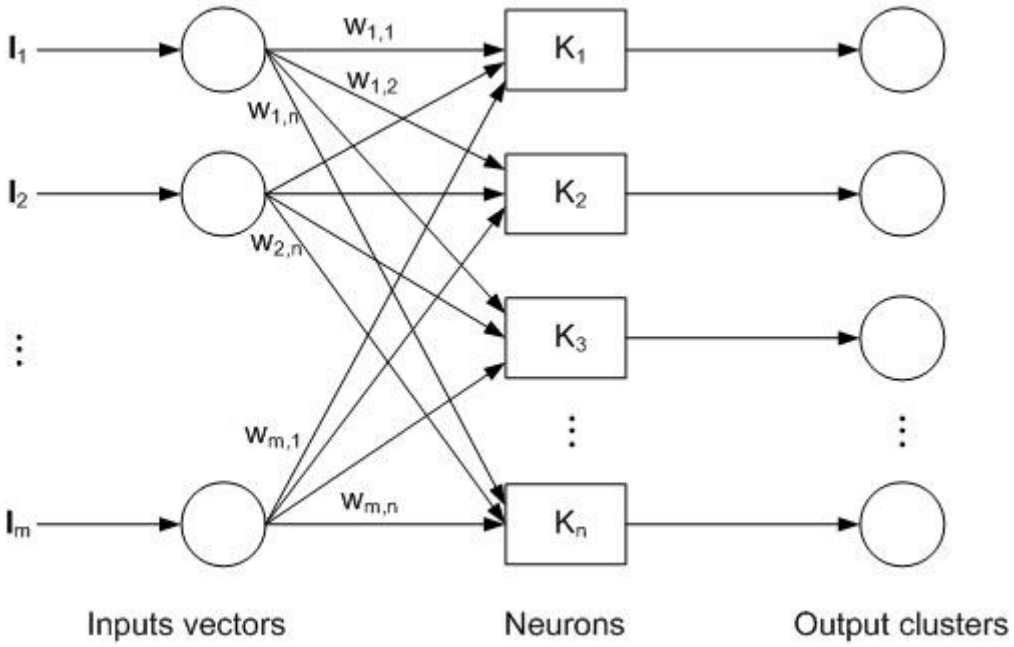
NORTHROP GRUMMAN

Possible features in a connection record

- **other features:**
  - Number of SYN packets sent from src to dst and vice versa
  - Number of ACK packets sent from src to dst and vice versa
  - Maximum segment size requested in the SYN packet by the src
  - Maximum segment size requested in the SYN packet by the dst
  - Total number of bytes sent in the initial window prior to receiving the first ACK packet from the destination.
  - Total number of bytes sent in the initial window prior to receiving the first ACK packet from the src.
  - Number of connections from the same source IP address to the same dst IP address
  - Number of connections that have SYN errors
  - Number of connections that have REJ errors

NORTHROP GRUMMAN

Winner-take-all Kohonen net (Simple K-map)



NORTHROP GRUMMAN

The K-map

- **One of the simplest neural network AKA self-organizing map**
- **Unsupervised neural network**
- **One competitive layer of neurons**
  - Each neuron has an associated weight vector
- **Input layer**
  - Feeds a set of input vectors to the neurons in the competitive layer
- **Output layer**
  - A set of clusters, one for each neuron
  - The group of input vectors that are closest to that neuron's weight vector.
- **The winner-take-all strategy**
  - Feed each input vector to each of the neurons
  - Compute dot product (net_i) of input vector and neuron's weight vector
  - Choose the best matching unit (the neuron with the maximum net_m).
  - Tag the input vector to neuron m's cluster.
- **Two phases: training phase (offline), detection phase**

NORTHROP GRUMMAN

Is K-map a good candidate for network anomaly detection?

- **Three essential metrics :**
  - Must be Computationally efficient
    - The detection phase should be fast.
  - Must give good detection rate.
    - catch as many of the anomalies as possible
  - Must give low number of false positives
    - The number of false identification of normal records as anomalous should be small.

NORTHROP GRUMMAN

Putting the k-map to the test

- **Training algorithm**

Step 1: Obtain the following inputs from the user.

- Number of neurons, *numNeurons*
- The size of the input vector, *numFeatures,* that is determined from the feature subset to use
- The *threshold* to determine the number of training iterations
- The names of input and output files, where input file represents the file in which the training data set is located and output file represents the file to which the trained state of the K-Map is stored.

Step 2: Read input records.

Step 3: Construct input vectors with the specified subset of features. Let *numVectors* denote the total number of vectors.

Step 4: Construct a single layer K-Map with the specified number of neurons (*numNeurons*) and the selected number of features (*numFeatures*). The K-Map will have a weight matrix of size *numNeurons* $\times$ *numFeatures*.

Step 5: Initialize the weight matrix with randomly selected input vectors and normalize the weight matrix.

Step 6: Choose an initial learning factor $\alpha 0 \leftarrow 1$.

Step 7: Compute *maxChanges* $\leftarrow$ *numVectors* $\times$ *threshold*

NORTHROP GRUMMAN

Training algorithm

Step 8: **Repeat**

$loopCount \leftarrow loopCount + 1$

$\alpha \leftarrow \alpha_0 / \sqrt{loopCount}; \quad numChanged \leftarrow 0$

For each input vector *I* do

Normalize I

For each row of the weight matrix, compute net_i

$$net_i \leftarrow \sum_{k=0}^{numFeatures-1} W_{i,k} * I_k$$

Choose the winner as the neuron *j,* where

$$net_j = \max(net_i), \ 0 \le i < numNeurons$$

Adjust the weights for the winner as

$$W_{winner,k} \leftarrow W_{winner,k} + \alpha * \left(I_k - W_{winner,k} \right), \ 0 \le k < numNeurons$$

Normalize the weight for the winner.

If the existing best matching unit for I, $map(I) \neq winner$

$$numChanged \leftarrow numChanged + 1$$

until $numChanged < \text{maxChanges}$

The detection phase

- **The trained K-map functions as a classifier**
- **Store the trained state of the NN in a file**
- **Test whether new connection records are normal or anomalous:**
  - Load the trained state into a K-map
  - Construct the feature vectors for the connection records.
  - Normalize each feature vector
  - Feed each vector to the input layer
  - Choose the winner neuron for that input
  - If the winner neuron represents a cluster that represents normal records, the input record represents a normal connection; otherwise it is an anomaly.

NORTHROP GRUMMAN

How good is a simple K-map in IDS?

- **Define detection rate:**
  - an instance of anomaly identified as normal, is a case of missed detection.
  - N_{attack} : the total number of attacks in the test set
  - N_{missed}: the number of missed instances
  - $\%Detected = (N_{attack} - N_{missed}) / N_{attack} * 100$
- **Define false positive rate:**
  - An instance of normal record falsely identified as anomaly is a false positive.
  - N_{normal} : number of normal records in the test set
  - N_{false} : total number of false positives
  - $\%FalsePositive = (N_{false} / N_{normal}) * 100$
- **Computational cost: (only detection phase is relevant)**
- **O(nk), where n is the number of input records, k is the number of neurons. As n tends to infinity, k is a constant, so in the traditional sense of computational complexity, it is a linear algorithm.**

NORTHROP GRUMMAN

Evaluating the K-map against a benchmark

- **The KDD 1998 and 1999 contests**
  - MIT Lincoln laboratory conducted IDS evaluation contests in 1998 and 1999 with DARPA sponsorship.
  - Simulated air force LAN environment was set up which generated normal traffic generated by 100's of users on 1000's of hosts.
  - Numerous attacks were carried out on this network
  - The network dump data was collected
  - Six weeks worth of training data and two weeks test data were supplied to the 6 research participants.
  - The test data contained new attack instances which were not present in the training data.
  - The results were analyzed for probe, denial of service, R2L and U2R
  - The best teams were able to detect between 63% to 93% of the old attacks that were present in the training data at about 10 false alarms per day.
  - Roughly half of the new attacks went undetected.
  - It was recommended that further research be conducted in finding new attacks rather than resorting to just extending rule-based approaches.

NORTHROP GRUMMAN

Benchmark data: The KDD Cup 1999 data

- **At the 3rd International Knowledge Discovery and Data mining Tools competition, S. Stolfo et al presented the KDD Cup 1999 benchmark data which consists of basic features and derived features from the original KDD99 dataset. (archived at the site http://kdd.ics.uci.edu/databases/kddCup99/kddCup99.html**
  - The training set and test set are organized as connection records in ASCII format.
  - For each connection record, there are 41 qualitative and quantitative features and a label indicating whether the record is normal or abnormal.
  - Separate sets of data are available for training and testing.
  - The records are predominantly attacks (80%) with about 39 different attacks representing all four categories of DOS, U2R, R2L and PROBE
  - 17 of these attack types are present only in test data

NORTHROP GRUMMAN

Training sets and test set

Type	KDD99 10%	Set 1	Set 2	Set 3	Set 4	Set 5	Test set1	Test set2
normal	97,278	25,000	8,600	8,740	97,278	20,676	60,593	60,593
neptune	107,201	5,000	25,000	9,491	45,700	22,783	58,001	58,001
smurf	280,790	5,000	9,610	24,971	17,270	0	164,091	0
back	2,203	2,203	201	192	2,203	0	1,098	0
satan	1,589	1,589	145	148	1,589	316	1,633	1,633
ipsweep	1,247	1,247	116	101	1,247	0	306	0
portsweep	1,040	1,040	93	106	1,040	225	354	354
warezclient	1,020	1,020	84	86	1,020		0	
teardrop	979	979	97	97	979		12	
pod	264	264	23	28	264		87	
nmap	231	231	16	29	231		84	
Guess_passwd	53	53	6	5	53		4,367	
Buffer_overflow	30	30	2	1	30		22	
Land	21	21	1	0	21		9	
warezmaster	20	20	4	0	20		1,602	
imap	12	12	1	0	12		1	
rootkit	10	10	0	2	10		13	
loadmodule	9	9	0	0	9		2	
ftp_write	8	8	0	1	8		3	
multihop	7	7	0	2	7		18	
phf	4	4	1	0	4		2	
perl	3	3	0	0	3		2	
spy	2	2	0	0	2		0	
snmpgetattack							7,741	
named							17	
xlock							9	
xsnoop							4	
sendmail							17	
saint							736	
apache2							794	
udpstorm							2	
mscan							1,053	
processtable							759	
ps							16	
worm							2	
httptunnel							158	
sqlattack							2	
snmpguess							2,406	
mailbomb							5,000	
xterm							13	
Total	**494,021**	**43,752**	**44,000**	**44,000**	**169,000**	**44,000**	**311,029**	**120,581**

NORTHROP GRUMMAN

Evaluation of K-Map with KDD Cup 99 data

- **Use all 41 features?**
  - May not be productive
  - Computationally costly
  - may not yield a reasonable grouping of the records
- **If so how do we base selection of subsets?**
  - Apply some domain knowledge from the field of IDS : Examples
    - The ICMP protocol, echo request service and the #bytes from src to destination can group smurf attack records into a cluster
    - One mode of ipsweep scan uses ICMP protocol and eco_i service to gather info on vulnerabilities. The number of bytes transmitted from source to destination is usually 8 bytes.
    - Nmap probe uses the port unreachable or Network unreachable ICMP messages from attempts to deliver UDP packets
    - Another form of nmap scan uses the half open SYN in TCP protocol.
    - Attacks such as teardrop exploit the weakness in the reassembly of packet fragments by creating fragments with overlapping offset fields.

NORTHROP GRUMMAN

Some feature subsets for the simple K-map

Feature subset 1:

- Protocol
  - tcp, udp, icmp
- Service
  - http, ftp, smtp, eco_i, ecr_i, …
- srcBytes
  - #bytes transmitted from source to destination
- dstBytes
  - #bytes transmitted from destination to source

NORTHROP GRUMMAN

Feature subset 2

- Protocol
- Service
- Duration (the length of the connection)
- srcBytes
- dstBytes
- wrongFragments (the number of wrong fragments encountered in reassembly)
- urgentPackets
- Count (the number of connections made to the same host in a given time interval)
- sameHstSynErrRate (fraction of connections that had SYN errors in a specified time interval)
- sameSvcSynErrRate (fraction of connections with the same host and same service that had SYN errors)
- sameHstRejErrRate (fraction of connections with the same host that had REJ errors)
- sameSvcRejErrRate (fraction of connections with the same host and same service that had REJ errors
- sameHstSameSvcRate (fraction of connections from the same host that used the same service)
- sameHstDiffSvcRate (fraction of connections from the same host that used different services)
- sameSvcDiffHstRate (fraction of connections that used the same service as the current connection that used different hosts in a given time interval)

NORTHROP GRUMMAN

Feature subset 3

- **features computed over 2 sec time window**
  - sameHstSynErrRate
  - sameSvcSynErrRate
  - sameHstRejErrRate
  - sameSvcRejErrRate
  - sameHstSameSvcRate
  - sameHstDiffSvcRate
  - sameSvcDiffHstRate

NORTHROP GRUMMAN

Tests and results for the simple K-map

- **Experiment 1:**
  - 36 neurons
  - all three feature subsets , three training data sets

Results:

Feature subset	Training data	%Detected	%False Positive
Feature set 1	Set 1	95.17	65.78
	Set 2	72.54	33.25
	Set 3	73.23	41.76
Feature set 2	Set 1	100	98.83
	Set 2	98.86	73.28
	Set 3	94.97	59.33
Feature set 3	Set 1	99.91	77.53
	Set 2	99.91	77.53
	Set 3	99.78	78.06

Experiment 2

- 48 neurons
- all three feature subsets , three training data sets
- Results:

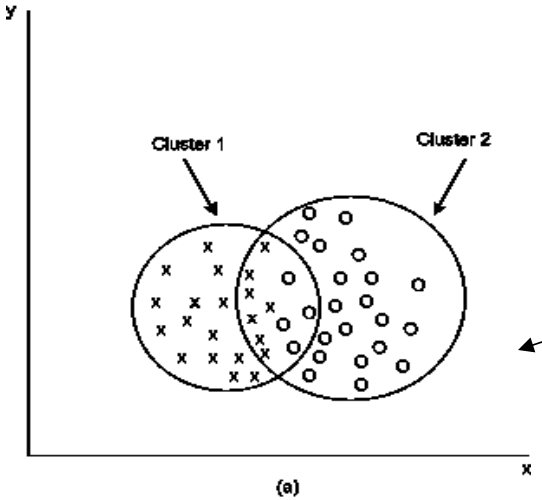
Feature Subset	Training Data	% Detected	%False Positive
Feature set 1	Set 1	98.21	82.32
	Set 2	72.53	32.18
	Set 3	73.23	40.14
Feature set 2	Set 1	100	98.69
	Set 2	86.71	62.21
	Set 3	94.05	62.35
Feature set 3	Set 1	99.9	76.39
	Set 2	99.52	75.47
	Set 3	99.7	76.85

Observations

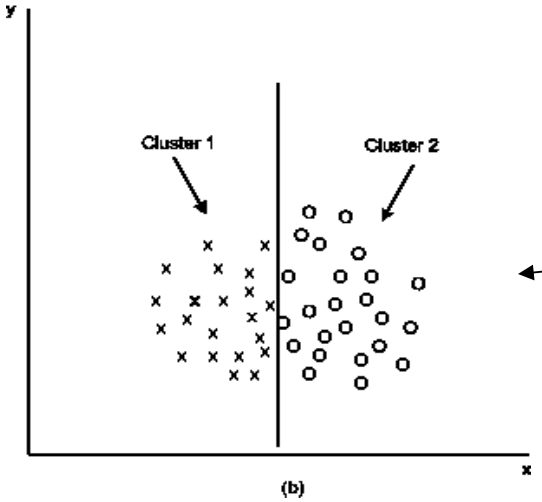
- **The simple K-map is not the ideal candidate**
  - Though detection rate is high, false positive rate is unacceptably high.
  - Increasing the number of neurons does not reduce the false positive rate.
  - Some feature subsets are more sensitive in detecting certain types of anomalies.
  - The single-layer K-Map is useful generally in the sense that it helps to group similar input vectors into clusters, however it does not guarantee optimal separation of resulting clusters.
  - Clusters can vary significantly depending on the dimension of the clustering boundary.

NORTHROP GRUMMAN

Effect of feature space dimension on clustering



Two-dimensional subspace mapping



One-dimensional subspace mapping

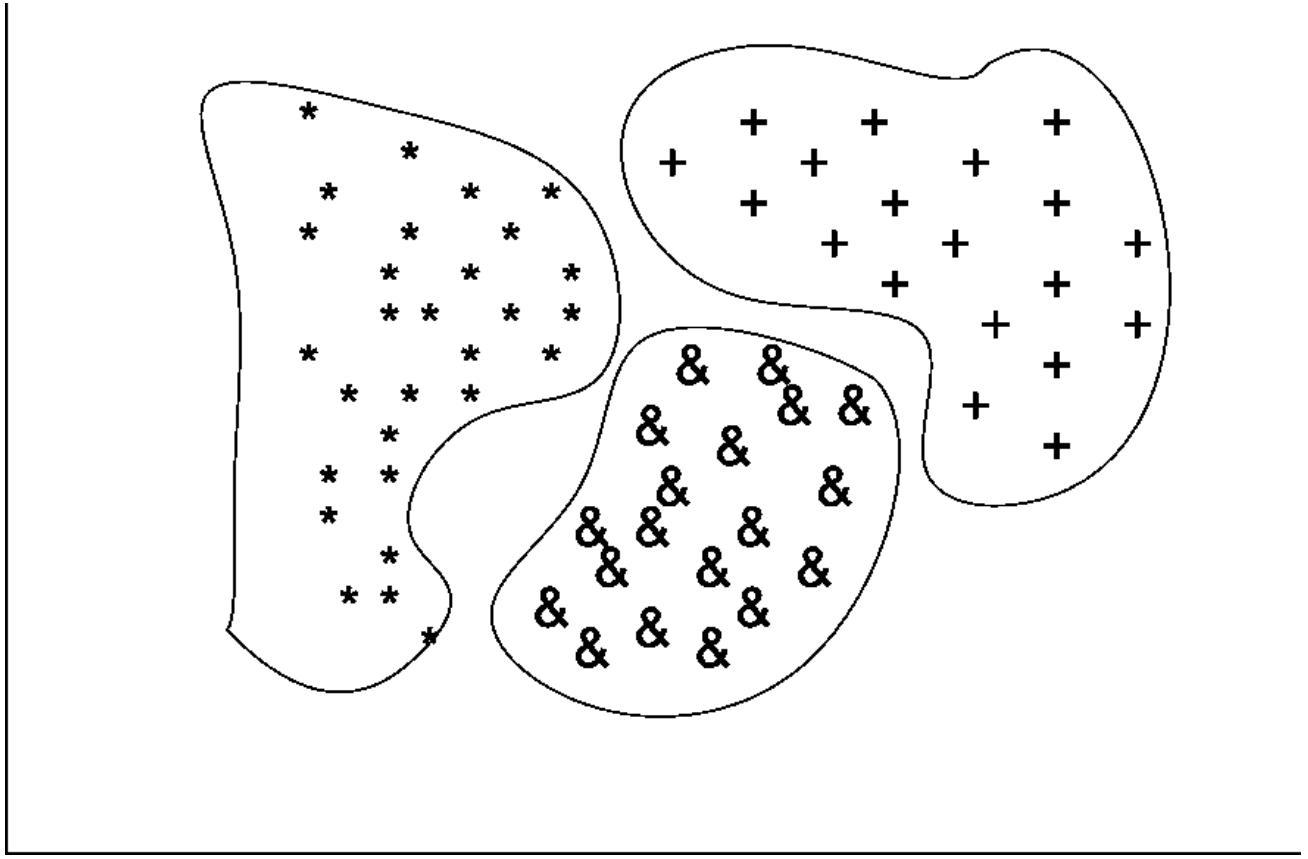
NORTHROP GRUMMAN

A hierarchical winner-take-all K-Map

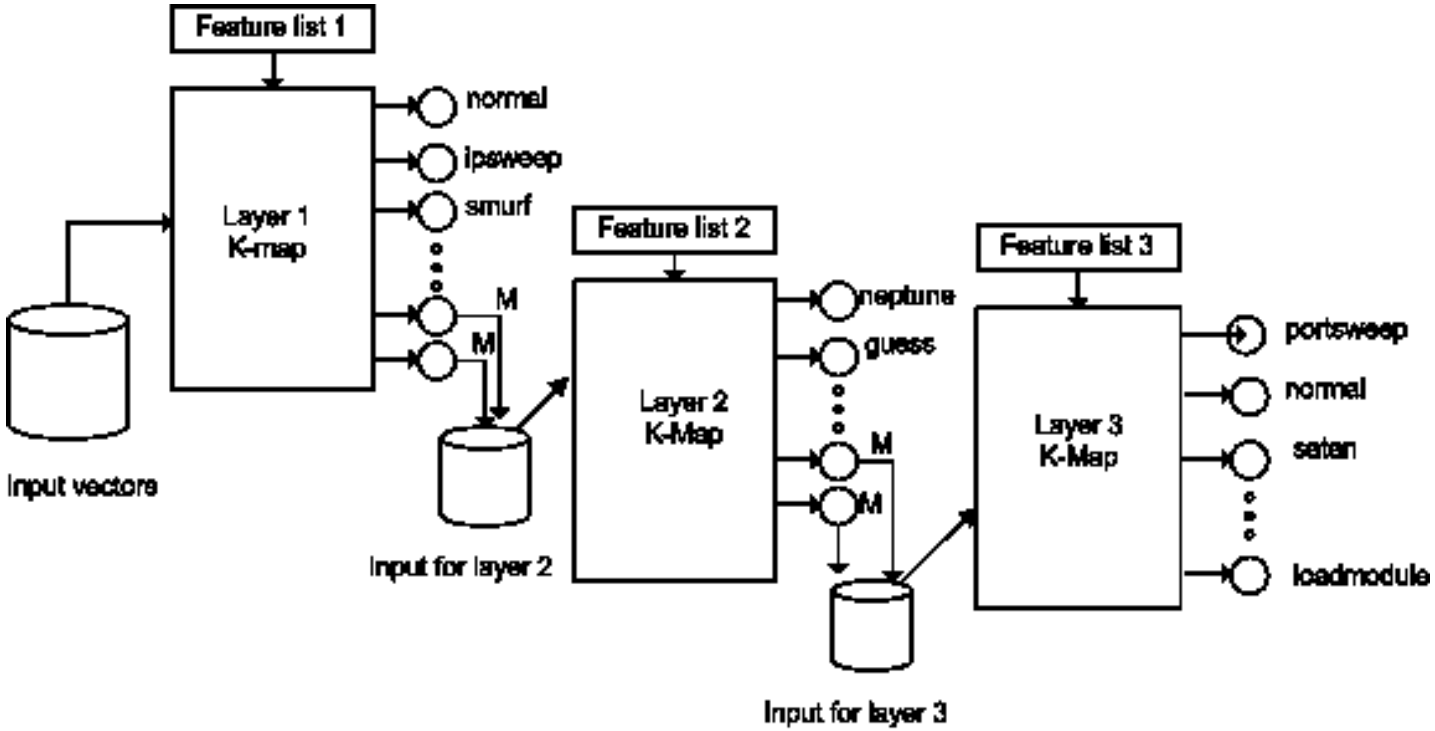
- **Motivation:**
  - The hyper-spherical clusters of simple K-map are insufficient. A high-order nonlinear classifier model is needed.
- **Technique:**
  - Select mutually exclusive sub-sets of the feature vector.
  - Feed the subsets to multiple neuron layers.
  - The resulting clusters model the intersection of hyper-cylinders.

NORTHROP GRUMMAN

Clusters formed by intersection of hyper-cylinders



A 3-level hierarchical winner-take-all map



M: A heterogeneous mix of attack records and/or normal records

NORTHROP GRUMMAN

Training algorithm

1: Get #levels, n from the user. Let Φ denote the set of input records, X_i the number of records of type X in cluster i, N_i the number of records in cluster i and X_{total} the total number of records of type X in the data set.

2: Extract the feature subspace for each of the n levels.

3: For each level s $\in$ {1, 2, …, n} repeat :

Train level s map with the training algorithm for simple K-map.

For each cluster i in level s,

If cluster i is homogeneous

label cluster i with the unique label of the records in cluster i

Set confidence factor $C_i \leftarrow 1.0$

Let H denote the set of records that mapped to i

$$\Phi \leftarrow \Phi - H$$

NORTHROP GRUMMAN

Training algorithm (contd.)

else begin

Compute A, the set of labels that mapped to i

For each label X in cluster i, compute

$F_i(X) = (X_i / X_{total})\ (X_i / N_i)$

Set label for cluster i, L(i) $\leftarrow \alpha$, where

$$F_i(\alpha) = \max_{x \in A} F_i(x)$$

set confidence factor

$$C_i = \frac{\max F_i(X)}{\sum_{X \in A} F_i(X)}$$

end (else)

4: store the trained state

---------- **End Training** ----------

A Definition: If a group of input feature vectors are being grouped to a neuron's cluster in level i with 100% confidence, then we consider that set reliably classified at level i.

Detection Algorithm for hierarchical K-map

Step 1: Initialize the hierarchical K-Map with the parameters stored in a state file.

Step 2: **for** each KDD 99 Cup test record **do**

Construct the test vectors for levels *1* to *n* using the corresponding feature subspace.

$reliablyClassified \leftarrow false$

$level \leftarrow 1$

while ($level < n$) **and not** $reliablyClassified$ **do**

Feed the test vector for level *i* to the i^{th} level K-Map.

Look up the encapsulation of label, neuron number, confidence etc for the winner neuron.

$reliablyClassified \leftarrow (confidence\ \ equals\ \ 1)\ \text{or}\ (label = \text{"Undefined"})$

$level \leftarrow level + 1$

end while

Step 3: Choose the label and corresponding confidence from the level that has highest confidence.

Step 4: Compute total number of false positives and detected anomalies

Some feature subset combinations

Combination	First Level Features	Second Level Features	Third Level Features
1	dstBytes srcBytes Protocol Service	Fragments Flag numRootAccess Logged status Hot Failed logins numFileCreation	sameSvcsynErrRate sameHstSameSvcRate sameHstDiffSvcRate sameSvcDiffHstRate dstHstSvcDiffHstRate dstHstSameSvcRate dstHstSameSrcPortRate Count sameHstSynErrRate
2	dstBytes srcBytes Protocol Fragments	Service Flag numRootAccess dstHstSameSvcRate dstHstSameSrcPortRate Count sameHstSynErrRate	sameSvcSynErrRate sameHstSameSvcRate sameHstDiffSvcRate sameSvcDiffHstRate dstHstSvcDiffHstRate Logged status Hot Failed logins numFileCreation
3	dstBytes srcBytes Protocol Service Fragments	Flag numRootAccess Logged status dstHstSameSvcRate dstHstSameSrcPortRate Count sameHstSynErrRate	Hot Failed logins numFileCreation sameSvcSynErrRate sameHstSameSvcRate sameHstDiffSvcRate sameSvcDiffHstRate dstHstSvcDiffHstRate sameHstRejErrRate

NORTHROP GRUMMAN

Feature set combinations (contd.)

4	dstBytes srcBytes Protocol Service Count	guestLogin Flag Hot Logged status dstHstSvcDiffHstRate dstHstSameSvcRate dstHstSameSrcPortRate Count sameHstSynErrRate	Failed logins numFileCreation numRootAccess sameSvcSynErrRate sameHstSameSvcRate sameHstDiffSvcRate sameSvcDiffHstRate sameHstRejErrRate Fragments
5	sameHstSynErrRate sameSvcSynErrRate sameHstRejErrRate sameSvcRejErrRate sameHstSameSvcRate sameHstDiffSvcRate SameSvcDiffHstRate	duration protocol service flag srcBytes dstBytes landStatus Fragments Urgent packets	hot Failed Logins numCompromised numRootAccess numFileCreation numShells numAccessFiles numOutBoundCommands

NORTHROP GRUMMAN

Test results for Hierarchical K-map

Experiment 1:

- 60 test cases using test set 1

Feature Combination	Training Set	% Anomaly Detection			% False Positive		
		36 neurons	48 neurons	72 neurons	36 neurons	48 neurons	72 neurons
Combination 1	Set 1	95.37	95.36	94.8	10.36	11.94	11.00
	Set 2	70.07	72.08	70.12	9.64	10.22	9.19
	Set 3	72.08	72.53	70.82	10.24	10.40	10.02
	Set 4	70.3	70.75	70.68	2.87	3.35	2.75
Combination 2	Set 1	90.73	87.74	3.99	4.87	4.92	9.61
	Set 2	94.7	94.39	24.72	27.09	24.74	1.22
	Set 3	95.11	95.02	24.93	27.38	24.38	3.24
	Set 4	92.02	92.56	90.94	3.63	3.00	3.43
Combination 3	Set 1	94.88	95.42	94.79	12.93	14.35	11.06
	Set 2	94.94	94.51	90.44	25.68	24.25	10.18
	Set 3	93.52	93.32	92.59	10.46	12.98	10.82
	Set 4	93.35	93.46	93.27	3.56	3.99	3.41
Combination 4	Set 1	47.69	47.78	77.86	2.49	2.40	1.52
	Set 2	93.26	93.05	60.69	19.62	17.16	3.04
	Set 3	91.97	91.55	77.45	6.84	7.66	2.00
	Set 4	78.18	91.37	91.29	2.19	2.92	2.76
Combination 5	Set 1	94.9	95.07	95.13	12.09	13.77	12.30
	Set 2	93.36	96.29	96.42	14.63	29.69	29.51
	Set 3	94.41	94.31	97.70	14.74	13.22	24.97
	Set 4	79.07	94.92	94.67	13.23	13.16	11.36

NORTHROP GRUMMAN

Test results (contd.)

- **Experiment 2: Limited attack types to just Neptune, ipsweep, and portsweep (training set 5 and test set 2) and tested for feature combinations 1 to 4.**

Feature combination	%Detected	%False positive
1	99.63	0.34
2	99.17	0.45
3	99.6	0.35
4	99.51	1.41

Experiment 3: Results for dump data obtained from STEAL lab

Event type	GHT	HTG
cisco	90.91%	90.91%
dos	97.27%	96.72%
finger_abuses	66.67%	66.67%
firewalls	90.00%	90.00%
ftp	89.66%	86.21%
Gain_root	92.86%	93.88%
Gain_shell	100.00%	100.00%
General	96.84%	95.79%
port_scanner	99.95%	99.95%
Remote_file_access	97.73%	97.73%
smtp_problem	92.86%	92.86%
backdoor	96.59%	96.59%
SNMP	100.00%	100.00%
P2P_FileSharing	**89.47%**	**89.47%**
Misc	**92.91%**	**92.91%**
NIS	**75.00%**	**75.00%**
RPC	**66.67%**	**66.67%**
Unix_account	**98.46%**	**98.46%**
Useless_service	**82.22%**	**77.78%**
windows	**30.85%**	**27.66%**
%FP	1.33%	1.33%
%overall detected	96.10%	95.87%

Shows percentage of attacks detected by type, **Previously unseen attacks in bold**

NORTHROP GRUMMAN

Observations – Hierarchical K-map

- Computationally efficient. Detection takes O(n) time.
- False positive rates are quite low compared to simple K-map and many other anomaly detection algorithms.
- Gives high detection rate
- Got detection rates between 90.94% and 93.46% at false positive rates between 2.19% and 3.99%.
- When attacks were limited to just 3, achieved 99.63% detection at 0.34% false positive rate.
- Training and testing was also conducted on network dump data other than KDD 99 set. The results were excellent.
- Looks like hierarchical K-map is an ideal candidate for anomaly detection.

NORTHROP GRUMMAN

Summary

1. **Anomaly detection**
    1. Definition
    2. General techniques
        1. Statistical
        2. Neural network
        3. Machine learning
2. **Intrusion detection**
    1. Categories
    2. Algorithmic techniques
3. **Anomaly detection techniques in IDS**
    1. A novel anomaly detection technique using Kohonen network
    2. Conclusions from the tests conducted on multi-level K-map

NORTHROP GRUMMAN

Some References

1. V. Barnett and T. Lewis, *Outliers in Statistical data*, John Wiley & sons, 3rd ed, 1994.
2. P. Rousseeuw, and A. Leroy, A, *Robust Regression and Outlier Detection*, John Wiley & sons., 3rd ed, 1996.
3. D. Dasgupta, and S. Forrest, Novelty Detection in Time Series Data using Ideas from Immunology, *Proc. of Fifth International Conference on Intelligent Systems*, 1996.
4. W. Lee, S. Stolfo, and K. Mok, "A data mining framework for building intrusion detection models", in *Proc. 1999 IEEE Symp. Security and Privacy*, 1999.
5. S. Northcut and J. Novak, *Network Intrusion Detection*, 3rd ed. Indianapolis, IN: New Riders Publishing, 2002.
6. S. Stolfo et al (2002) The Third International Discovery and Data Mining Tools Competition. [online], Available: http://kdd.ics.uci.edu/database/kddCup99/kddCup99.html
7. T. Kohonen, *Self-Organizing Maps*, 3rd extended ed, ser. Information Sciences, Berlin, Germany: Springer, vol. 30, 2001.
8. S. Sarasamma, Q. Zhu, J. Huff, "Hierarchical Kohonen Net for Anomaly Detection in Network Security", *IEEE Transactions on Systems, Man and Cybernetics - part B*, vol. 35, No. 2, 2005.

NORTHROP GRUMMAN

Appendix

- **Distance metrics**

NORTHROP GRUMMAN

Distance metrics - Appendix

- **Euclidean Distance**

$$X = (x_1, x_2, ..., x_n) \; ; \qquad Y = (y_1, y_2, ..., y_n)$$

$$d = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

- **Mahalanobis distance**

$$d = \sqrt{(X - \mu)^T \, \Sigma^{-1} \, (X - \mu)}$$

NORTHROP GRUMMAN