**Chapter 4**

**Propose work and methodology**

**4.1 PROPOSED WORK**

An intrusion can be defined as an action aimed at compromising the security requirements such as confidentiality, integrity or availability of data. This includes unauthorized attempts to access data, manipulate data or make the system not viable [7]. An algorithm aimed to detect the outliers shared by a networked organization has been proposed to detect the intrusion. Outliers are usually small clusters and the goal is to use outlier lists from different systems (based on a similar clustering, involving the same similarity measure). If an outlier occurs for at least two systems, then it is considered as an attack based on the assumption that an intrusion attempt that tries to find a weakness of a script will look similar for all the victims of this attack. Once the intrusion has been detected successfully then the administrator can properly set up a network to be more secure.

An algorithm is applied, to perform a clustering on the usage patterns of each site and to find the common outliers. The first step for clustering the patterns of each site is to find the similarity between the patterns. The similarity measure (presented in section 3.2) will allow normal usage patterns to be grouped together and distinguishes an intrusion pattern from normal usage patterns and from other intrusion patterns (since different intrusion patterns will be based on a different security hole and will have very different characteristics). The algorithm performs successive clustering for each site. At each step we check the potentially matching outliers between both sites. The clustering algorithm is agglomerative and depends on the similarity measure respected between two objects. Then, the alarms will be triggered at each step of the monitoring (for instance for every one hour). The assumption is that common outliers, sorted by similarity from one site to another, will be added to the intrusions list.

**4.2 methodology**

**METHODOLOGY:1**

The SHA-1 Hash encryption algorithm specifies a Secure Hash Algorithm, which can be used to generate a condensed representation of a message called a message digest. The algorithm is required for use with the Digital Signature Algorithm (DSA) as specified in the Digital Signature Standard (DSS) and whenever a secure hash algorithm is required. Both the transmitter and intended receiver of a message in computing and verifying a digital signature uses this method.

The same SHA-1 algorithm, but employing a variable key size, is used to create the SHA-2 family of functions. The four hash functions that comprise SHA-2 are SHA-224, SHA-256, SHA-384, and SHA-512, with the numeric portion of the name indicating the number of bits in the key. SHA-2 functions are more secure than SHA-1 although not as widely used currently.

SHA-1 Hash is used for computing a condensed representation of a message or a data file. When a message of any length < 2 64 bits is input, the Hash algorithm produces a 160-bit output called a message digest. The message digest can then be input to the Digital Signature Algorithm (DSA), which generates or verifies the signature for the message. Signing the message digest rather than the message often improves the efficiency of the process because the message digest is usually much smaller in size than the message. The same hash algorithm must be used by the verifier of a digital signature as was used by the creator of the digital signature.

The SHA-1 Hash is called secure because it is computationally infeasible to find a message which corresponds to a given message digest, or to find two different messages which produce the same message digest. Any change to a message in transit will, with very high probability, result in a different message digest, and the signature will fail to verify. SHA-1 is a technical revision of SHA (FIPS 180). A circular left shift operation has been added to the SHA (FIPS 180). SHA-1 improves the security provided by the SHA standard. The SHA-1 is based on principles similar to those used by the MD4 message digest algorithm.

**Features:-**

\* The algorithm is used to compute a message digest for a message or data file that is provided as input.

\* The message or data file should be considered to be a bit string.

\* The length of the message is the number of bits in the message (the empty message has length 0).

\* If the number of bits in a message is a multiple of 8, for compactness we can represent the message in hex.

\* The purpose of message padding is to make the total length of a padded message a multiple of 512.

\* The purpose of message padding is to make the total length of a padded message a multiple of 512.

\* As a summary, a “1″ followed by m “0″s followed by a 64-bit integer are appended to the end of the message to produce a padded message of length 512 \* n.

\* The 64-bit integer is l, the length of the original message.

\* The padded message is then processed by the SHA-1 as n 512-bit blocks.

**METHODOLOGY:2**

**Clustering algorithms** are either of type partitioned or hierarchical methods. The algorithms studied on clustering of categorical sequences [6, 11] use an edit distance or sequence alignment method for finding the similarity between sequences. An agglomerative hierarchical clustering algorithm [13] is used here for clustering sequences. Consider the problem of clustering n sequences of characters. First, each of the (n) x (n-1)/2 pairs of possible merges is evaluated, and the two clusters that have maximum value of the criterion function are merged. After performing m merging steps, each of the (n-m) x (n-m-1)/2 pairs possible merges is evaluated. This process continues until there are only k clusters left. The criterion function [13] used is Maximize Cf = cj,ik1rr)j,i(sim n 1)where nr is the number of sequences in Cr and k is the number of clusters. The method used for clustering the remaining sequence data is the k-nearest-neighbor (k-nn) method. This method merges International Journal of Computer Applications (0975 – 8887) Volume 30– No.4, September 2011.

A new sequence with one of the generated clusters by computing the similarity between the new sequence and sequences of the clusters and finds a cluster having the most k-nearest neighbors out of it. If an equal number of k-nearest-neighbors exists for more than one cluster, choose one cluster randomly.