

A graph theoretical model of computer security

From file sharing to social engineering

Mark Burgess¹, Geoffrey Canright², Kenth Engø-Monsen²

¹ Faculty of Engineering, Oslo University College, Norway

² Telenor Research, Fornebu, Oslo, Norway

26. May 2003

Abstract We describe a model of computer security that applies results from the statistical properties of graphs to human-computer systems. The model attempts to determine a safe threshold of interconnectivity in a human-computer system by ad hoc network analyses. The results can be applied to physical networks, social networks and to networks of clues in a forensic analysis. Access control, intrusions and social engineering can also be discussed as graphical and information theoretical relationships. Groups of users and shared objects, such as files or conversations, provide communications channels for the spread of both authorized and un-authorized information. We present numerical criteria for measuring the security of such systems and algorithms for finding the vulnerable points.

Keywords: Social networks, access control, social engineering, forensic analysis.

1 Introduction

File access control is one of the less glamorous aspects of system security, but it is the most basic defence in the containment of information. While a technology like encryption has high-profile credibility for security, it is still nothing more than ‘security through obscurity’ and its real effectiveness lies in the management of access to its basic secrets (i.e. the encryption keys). Security is a property of an *entire system* including explicit and covert channels. Unexpected routes such as social channels, are often used to circumvent so-called strong security mechanisms. The file security problem is a generic representation of communication flow around a system, and thus we return to this basic problem to ask whether something more quantitative can be said about it. The issue of social engineering has previously been rather difficult to address.

Graph theoretical methods have long been used to discuss issues in computer security[1–3]. Typically graphs have been used to discuss trust

relationships and restricted information flows (privacy). To our knowledge, no-one has considered graphical methods as a practical tool for performing a partially automated analysis of real computer system security. Computer systems can form relatively large graphs. The Internet is perhaps the largest graph that has ever been studied, and much research has been directed at analyzing the flow of information through it. Research shows that the Internet[4] and the Web[5] (the latter viewed as a directed graph) each have a power-law degree distribution. Such a distribution is characteristic[6–8] of a self-organized network, such as a social network, rather than a purely technological one. Increasingly we see technology being deployed in a pattern that mimics social networks, as humans bind together different technologies, such as the Internet, the telephone system and verbal communication.

Social networks have many interesting features, but their most interesting feature is that they do not always have a well defined centre, or point of origin; this makes them highly robust to failure, and also extremely transparent to the percolation of both ‘good’ and ‘bad’ information[9]. A question of particular interest to computer security is: can we identify likely points of attack in a general network of associations, and use this information to build analytical tools for securing human-computer systems? Users are related to one another by various associations: file sharing, peer groups, friends, message exchange, etc. Every such connection represents a potential information flow. An analysis of these can be useful in several instances:

- For finding the weakest points of a security infra-structure for preventative measures.
- In forensic analysis of breaches, to trace the impact of radiated damage at a particular point, or to trace back to the possible source.

There is scope for considerable research in this area. We begin with somewhat modest intentions and try to answer the simplest questions in a quantitative fashion, such as how does one use the properties of a graph to make characterizations about a system that would be of interest in a security context (see the plan in the next two paragraphs). How can we estimate the probable risk associated with a system purely from a topological viewpoint, using various models of exposure? We use the observation that human-computer communication lies at the heart of security breaches. We diverge from other discussions by further noting that communication takes place over many channels, some of which are controlled and others that are *covert*. A covert channel is a pathway for information that is not intended for communicating information and is therefore not usually subject to security controls, i.e. a security leak. Thus, our discussion unifies machine controllable security measures with measures for addressing social engineering, which usually resist analysis.

The plan for this paper is as follows: we begin by describing the layout of an organization, as a bipartite graph of file-like objects and users, though we shall occasionally disregard the bipartite nature for simplicity. We then define the meaning of collaborative groups, and find a shorthand

notation for visualizing the potential damage that might be spread by virtue of the connections within the organizational graph. In section 4, we consider the groups of nodes as being independent—ignoring any overlaps between them—but immerse them in a bath of possibly hostile information. Using information theory, we find a condition for maximizing the robustness of the system to outside exposure (i.e. risk of leaks or intrusions) and find that some groups are exponentially more important than others to security.

In section 5, we admit the possibility of links between groups, and consider percolation of data through the network. That is, we ask: is there a critical level of interconnectivity, above which information can flow essentially unchecked through the entire system? Finally, in section 6, we focus on fully-connected systems and consider which nodes in a graph are the most important (central) to the spread of information.

2 Associative bipartite graphs

The basic model one has is of a number of *users*, related by associations that are mediated by human-computer *resources*. The graphs we discuss in this paper represent a *single organization*. We do not draw any nodes for outsiders; rather we shall view outsiders as a kind of reservoir of potential danger in which our organization is immersed.

In the simplest case, we can imagine that users have access to a number of files. Overlapping access to files allow information to be passed from user to user: this is a channel for information flow. For example, consider a set of F files, shared by U users (fig. 1).

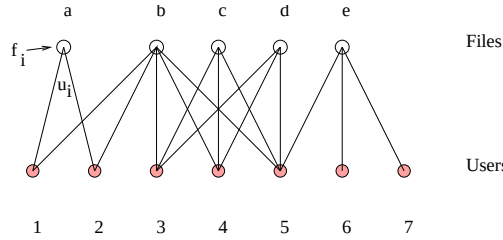


Fig. 1 Users (dark spots) are associated with one another through resources (light spots) that they share access to. Each light spot contains f_i files or sub-channels and defines a group i , through its association with u_i links to users. In computer parlance, they form ‘groups’.

Here we see two kinds of object (a bipartite graph), connected by links that represent associations. In between each object of one type must be an object of the other type. Each association is a potential channel for communication, either implicitly or explicitly. Dark spots represent different users, and light spots represent the files that they have access to. A file, or set

of files, connected to several users clearly forms a *system group*, in computer parlance. In graph-theory parlance the group—since it includes all possible user-file links—is simply a *complete (bipartite) subgraph*, or bipartite *clique*.

In Figure 2 we present another visual representation of the user/file bipartite graph, in the form of Venn diagrams. Here the users are emphasized, and the files are suppressed; the ellipses themselves then show the group structure. Leakage of information (eg damage) can occur between any groups having overlap in the Venn-diagram picture.

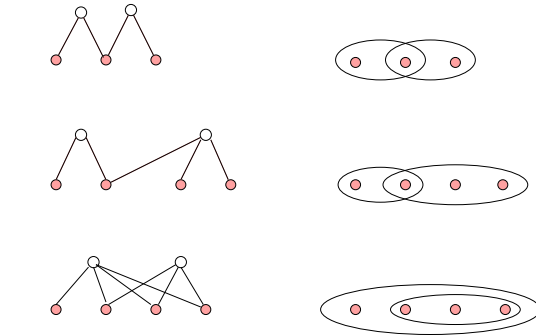


Fig. 2 Some example bipartite graphs drawn in two forms. On the right hand side of the diagram, the Venn diagrams acts as topographical contour lines, indicating the relative connectivity of the nodes. This is a useful graphical shorthand: an overlapping ellipse represents a potential leak out of the group

In reality, there are many levels of association between users that could act as channels for communication:

- Group work association (access).
- Friends, family or other social association.
- Physical location of users.

In a recent security incident at a University in Norway, a cracker gained complete access to systems because all hosts had a common root password. This is another common factor that binds ‘users’ at the host level, forming a graph that looks like a giant central hub. In a *post factum* forensic investigation, all of these possible routes of association between possible perpetrators of a crime are potentially important clues linking people together. Even in an *a priori* analysis such generalized networks might be used to address the likely targets of social engineering. In spite of the difference in principle of the network connections, all of these channels can be reduced to effective intermediary nodes, or meta-objects like files. For the initial discussion, at least, we need not distinguish them.

Each user naturally has a number of file objects that are private. These are represented by a single line from each user to a single object. Since

all users have these, they can be taken for granted and removed from the diagram in order to emphasize the role of more special hubs (see fig. 3).

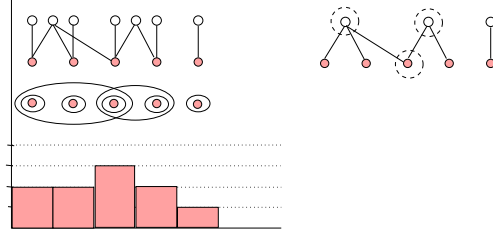


Fig. 3 An example of the simplest level at which a graph may be reduced to a skeleton form and how hot-spots are identified. This is essentially a renormalization of the histogram, or ‘height above sea-level’ for the contour picture.

The resulting contour graph, formed by the Venn diagrams, is the first indication of potential hot-spots in the local graph topology. Later we can replace this with a better measure — the ‘centrality’ or ‘well-connectedness’ of each node in the graph.

Bipartite graphs have been examined before to provide a framework for discussing security[10]. We take this idea a step further by treating the presence of links probabilistically.

3 Graph shorthand notations

The complexity of the basic bipartite graph and the insight so easily revealed from the Venn diagrams beg the question: is there a simpler representation of the graphs that summarizes their structure and which highlights their most important information channels?

An important clue is provided by the Venn diagrams; indeed these suffice to reveal a convenient level of detail in simple cases. We begin by defining the lowest level of detail required, using the following terms:

Definition 1 (Trivial group) *An ellipse that encircles only a single user node is a trivial group. It contains only one user.*

Definition 2 (Elementary group) *For each file node i , obtain the maximal group of users connected to the node and encircle these with a suitable ellipse (as in fig. 3). An ellipse that contains only trivial groups, as subgroups, is an elementary group.*

Our aim in simplifying a graph is to eliminate all detail within elementary groups, but retain the structure of the links between them. This simplification is motivated by our interest in damage control: since each group is a complete subgraph, it is natural to assume that intra-group infection occurs

on a shorter timescale than intergroup infection. Hence a natural coarsening views the elementary group as a unit.

Definition 3 (Simplification rule) *For each file node i , obtain the maximal group of users connected to the node and encircle these with a suitable ellipse or other envelope, ignoring repeated ellipses (as in fig. 3). Draw a super-node for each group, labelled by the total degree of group (the number of users within it). For each overlapping ellipse, draw an unbroken line between the groups that are connected by an overlap. These are cases where one or more users belongs to more than one group, i.e. there is a direct association. For each ellipse that encapsulates more than one elementary groups, draw a dashed line (see fig. 4).*

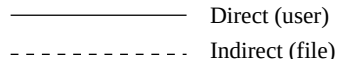


Fig. 4 Solid lines in the short hand denote direct user connections, formed by overlapping groups in the Venn diagram, i.e. users who belong to more than one group directly. Dotted lines represent associations that are formed by the envelopment of several subgroups by a parent group. This occurs when two groups of users have access to one or more common files.

The simple examples in fig. 2 can thus be drawn as in fig. 5. Our aim here is not to develop a one to one mapping of the graph, but to eliminate detail. There is no unique prescription involved here, rather one is guided by utility.

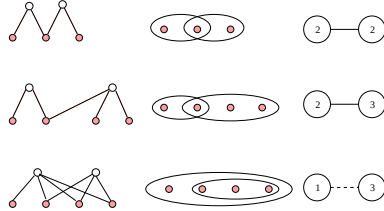


Fig. 5 Identification of elementary groups in example graphs of fig. 2

Further elimination of detail is also possible. For example, non-elementary groups are simply complete subgraphs plus some extra links. Hence it may be useful to view non-elementary groups as units. This kind of coarsening gives, for the last example in fig. 5, a single circle with group degree 4.

As a further example, we can take the graph used in ref. [11]. Fig. 6 shows the graph from that reference. Fig. 7 shows the same graph after eliminating the intermediate nodes. Finally, Figs. 8 and 9 show this graph

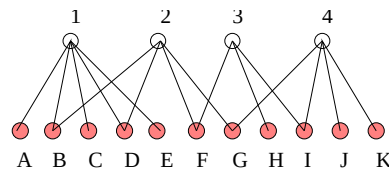


Fig. 6 The example bipartite graph from ref. [11] serves as an example of the shorthand procedure.

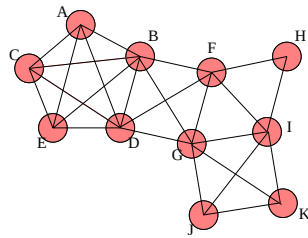


Fig. 7 A ‘one-mode’ projection of the graph in fig. 6, as given by ref. [11] is formed by the elimination of the intermediary nodes. Note that bipartite cliques in the original appear here also as cliques.

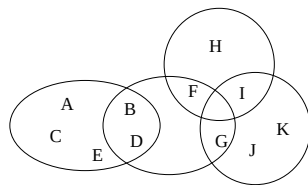


Fig. 8 The Venn diagram for the graph in fig. 6 shows simple and direct associations that resemble the one-mode projection, without details.

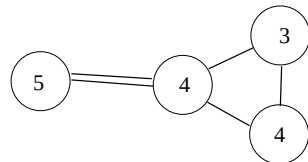


Fig. 9 The final compressed form of the graph in fig. 6 eliminates all detail but retains the security pertinent facts about the graph.

in our notation (respectively, the Venn diagram and the elementary-group shorthand).

The shorthand graphs (as in Fig. 9) may be useful in allowing one to see more easily when a big group or a small group is likely to be infected by bad information.

4 Information content and group exposure

Intuition tells us that there ought to be a reasonable limit to the number of collaborative groups on a system. If every user belongs to a number of groups, then the number of possibilities for overlap quickly increases, which suggests a loss of system boundaries. If an element of a group (a file or user) becomes corrupted or infected, that influence will spread if boundaries are lost. Clearly, we must quantify the importance of groups to security, and try to elucidate the role of group size and number.

4.1 Informational entropy

The first step in our sequence of steps is to consider a number of isolated groups that do not intercommunicate, but which are exposed to an external bath of users (e.g. the Internet), of which certain fraction is hostile. We then ask the question, how does the size of a group affect its vulnerability to attack from this external bath of users?

The maximum number of possible groups $G_{\max}$, of any size, that one could create on a system of U users is

$$G_{\max} = 2^U - 1. \quad (1)$$

That is, each group is defined by a binary choice (member or not) on each of U users—but we do not count the group with no members. Groups that do not fall into this basic set are aliases of one another. This number is still huge.

There are clearly many ways of counting groups; we begin by considering the problem of non-overlapping groups, i.e. only non-overlapping subgraphs (partitions). Groups are thus (in this section) assumed to be isolated from one another; but all are exposed to the (potentially hostile) environment. Given these conditions, we then ask: Is there a way to decide how many users should be in each group, given that we would like to minimize the exposure to the system?

Let $i = 1 \dots G$, where G is the number of groups defined on a system. In the following, we will treat G as fixed. Varying G , while maximizing security, generally gives the uninteresting result that the number of groups should be maximized, i.e., equal to the number of users—giving maximal security, but no communication. This is analogous to the similar result in ref. [12].

Let u_i be the number of users encapsulated in group i , and f_i be the number of files in (accessible to) group i . We say that a link belongs to group i if both its end nodes do. The total number of links in a group is then

$$L_i = u_i f_i, \quad (2)$$

(see fig. 1) and the probability that a given link is in group i is thus

$$p_i = \frac{L_i}{\sum_{i=1}^G L_i}. \quad (3)$$

Now we can think of putting together a user/file system 'from scratch', by laying down links in some pattern. In the absence of any information about the different groups (other than their total number G) we want to treat all groups equally. For a large system, we can then lay down links *at random*, knowing that the resulting distribution of links will be roughly uniform. Hence we can (statistically) avoid large groups (and hence damage spreading to a large number of users and files), simply by maximizing the entropy (randomness) of the links.

The entropy of the links is

$$S = - \sum_i^G p_i \ln p_i. \quad (4)$$

With no constraints, maximizing the entropy leads to a flat distribution, i.e. that all the p_i should be equal in size, or

$$u_i \propto \frac{1}{f_i}, \quad (5)$$

i.e. the total number of links per group should be approximately constant. This clearly makes sense: if many users are involved, the way to minimize spread of errors is to minimize their access to files. However, this simple model does not distinguish between groups other than by size. Since there is no intergroup communication, this is not an interesting result. To introduce a measure of the security level of groups to outside penetration, we can introduce a number ϵ_i which represents the *exposure* of the files and users in group i to outside influences. ϵ_i is a property of group i that is influenced by security policy, and other vulnerability factors. That is, in our model, the ϵ_i cannot be varied: they are fixed parameters determined by 'outside' factors.

We are thus now looking at a closed model for an open system. We label each group i with a positive value ϵ_i that is zero for no exposure to outside contact, and that has some non-zero, positive value for increasing exposure. The value can be measured arbitrarily and calibrated against a scale β , so that $\beta\epsilon_i$ has a convenient value, for each i . The average exposure of the system is fixed by $\langle\epsilon\rangle$, but its overall importance is gauged by the scale β . β may thus be thought of as the 'degree of malice' of the bath of outsiders.

We will again maximize entropy to minimize risk. Typically, the properties of the outside bath are not open to control by the system administrator, who first makes the ϵ_i as low as possible, and then varies the group size, as given by the p_i , at fixed β . This gives the average exposure:

$$\sum_{i=1}^G p_i \epsilon_i = \langle \epsilon \rangle = \text{const.} \quad (6)$$

However, using a continuum approximation that is accurate for large G , both β and $\langle \epsilon \rangle$ can be taken to be smoothly variable, and an equivalent procedure is to fix the average exposure $\langle \epsilon \rangle$, then vary β . The result is the invertible function $\beta(\langle \epsilon \rangle)$, determined by the group sizes p_i . We therefore look for the distribution of p_i with maximum entropy, i.e. the configuration of minimal risk, that satisfies this exposure constraint, and the normalization constraint $\sum_i p_i = 1$. We use the well-known Lagrange method on the link-entropy Lagrangian:

$$\begin{aligned} \mathcal{L} = & - \sum_i^G p_i \ln p_i - A \left(\sum_i^G p_i - 1 \right) \\ & - \beta \left(\sum_i^G p_i \epsilon_i - \langle \epsilon \rangle \right). \end{aligned} \quad (7)$$

Maximizing $\mathcal{L}$ with respect to p_i, A, β , gives

$$\begin{aligned} p_i &= e^{-A-1} e^{-\beta \epsilon_i} \\ &= \frac{e^{-\beta \epsilon_i}}{\sum_{i=0}^G e^{-\beta \epsilon_i}}. \end{aligned} \quad (8)$$

Let us recall the context, and hence significance, of this result. We assume noninteracting groups, with each group i having a (given) degree of exposure ϵ_i to damage from attack. We then minimize the average exposure (where the average is taken over all groups) by maximizing the entropy of the links. Minimizing the average exposure is of course the best security goal one can hope for, given these constraints. Following this simple procedure gives us the result [Eq. 8] that groups with high exposure to attack should have exponentially fewer links than those with less exposure.

We can rephrase this result as follows:

Result 1 *Groups that have a high exposure are exponentially more important to security than other groups. Thus groups such as those that include network services should be policed with maximum attention to detail.*

Thus, on the basis of rather general arguments, in which we view the many groups as noninteracting, we arrive at the interesting conclusion that individual nodes with high exposure can have significantly higher importance to security than others. More specifically, equation (8) implies that the relative size of two groups i and j should be exponential in the quantity $\beta(\epsilon_i - \epsilon_j)$; thus if typical values of this latter quantity are large, there is strong pressure to hold the more highly-exposed groups to a small size.

This 'ideal-gas' model of the set of groups presumably retains some relevance when the interactions (communication) between groups is small, but nonzero. In this limit, damage spreading between groups takes place (as does energy transfer in a real gas), but the independent-groups approximation may still be useful. In the next section, we go beyond the assumption that intergroup interaction is weak, and view explicitly the spread of information within the graph.

5 Percolation in the graph

The next step in our discussion is to add the possibility of overlap between groups, by adding a number of channels that interconnect them. These channels can be files, overlapping user groups, mobile-phone conversations, friendships or any other form of association. The question then becomes: how many channels can one add before the system becomes (essentially) completely connected by the links? This is known as the percolation problem; and the onset, with increasing number of links, of (nearly) full connectivity is known as the formation of a *giant cluster* in the graph.

In classical percolation studies the links are added at random. In this section we will assume that the links are added so as to maintain a given *node degree distribution* p_k . That is, p_k is the fraction of nodes having degree k .

Definition 4 (Degree of a node) *In a non-directed graph, the number of links connecting node i to all other nodes is called the degree k_i of the node.*

In the next subsection we will take a first look at percolation on graphs by considering a regular, hierarchical structure. In the subsequent subsections, we will follow Ref. [11] in looking at "random graphs with node degree distribution p_k ". The phrase in quotes refers to a graph chosen at random from an ensemble of graphs, all of whom have the given node degree distribution. That is, the links are laid down at random, subject to the constraint coming from the degree distribution.

5.1 Percolation in a hierarchy

One of the simplest types of graph is the hierarchical tree. Hierarchical graphs are not a good model of user-file associations, but they are representative of many organizational structures. A very regular hierarchical graph in which each node has the same degree (number of neighbours) is known as the Cayley tree. Studies of percolation phase transitions in the Cayley model can give some insight into the computer security problem: at the 'percolation threshold' essentially all nodes are connected in a 'giant cluster'—meaning that damage can spread from one node to all others. For link density (probability) below this threshold value, such wide damage spreading cannot occur.

The Cayley tree is a reasonable approximation to a random network below its percolation point, since in the non-percolating network loops have negligible probability [9]. Thus we can use this model to estimate the threshold probability itself.

The Cayley tree is a loopless graph in which every node has z neighbours. The percolation model envisions an ideal 'full' structure with all edges present (in this case, the Cayley tree with $z = z_{CT}$ edges for each node), and then fills those edges with real links, independently with probability p . Since z_{CT} is the ideal number of links to neighbours, the condition [9] for a path to exist from a distant region to a given node is that at least one of the node's $(z_{CT} - 1)$ links in the tree is connected; i.e. the critical probability for connectivity p_c is defined through

$$(z_{CT} - 1)p_c \geq 1. \quad (9)$$

When

$$p > p_c, \quad (10)$$

the network is completely traversable from any node or, in a security context, any node has access to the rest of the nodes by some route. Although this model is quite simplistic, it offers a rough guideline for avoiding percolation in real computer systems.

To apply this to a more general network, we need to view the general graph as partial filling of a Cayley tree. Assuming the graph is large, then some nodes will have completely filled all the links adjacent to them in the ideal tree. The coordination number of these nodes is then z_{CT} . These nodes will also have the maximum degree (which we call $K_{\max}$) in the graph. That is, we take

$$z_{CT} = K_{\max}. \quad (11)$$

The critical probability of a link leading to a giant cluster is then (in this approximation)

$$p_c \sim \frac{1}{K_{\max} - 1}. \quad (12)$$

Now we must estimate the fraction p of filled links. Assuming (again) that our real graph is a partial filling of a Cayley tree, p is then the fraction of the links on the Cayley tree that are actually filled. For the bipartite case, we have F files and U users, and so the number of links in the real graph is $L = \frac{\langle z \rangle}{2}(U + F)$. The Cayley tree has $\frac{z_{CT}}{2}(U + F)$ links. Hence p is the ratio of these:

$$p = \langle z \rangle / z_{CT}. \quad (13)$$

(This also applies for the unipartite case.) The average neighbour number $\langle z \rangle$ may be calculated from the real in the usual ways. For instance, if one has the node degree distribution p_k , then $\langle z \rangle = \langle k \rangle = \sum_k k p_k$.

The above assumptions give the following rough guide for limiting damage spreading in a computer system.

Result 2 *A risk threshold, in the Cayley tree approximation, is approached when a system approaches this limit:*

$$p = \frac{\langle z \rangle}{K_{\max}} > p_c = \frac{1}{K_{\max} - 1} \quad (14)$$

or

$$\langle z \rangle > \frac{K_{\max}}{K_{\max} - 1} . \quad (15)$$

That is (again), we equate a security risk threshold with the percolation threshold. Below the percolation threshold, any damage affecting some nodes will not be able to spread to the entire system; above this threshold, it can. Result 15 gives us an *approximation* for the percolation threshold. It is extremely simple and easy to calculate; but, being based on modeling the graph as a hierarchical Cayley tree, it is likely to give wrong answers in some cases.

5.2 Large random unipartite networks with arbitrary degree distributions

User-file and other security associations are not typically hierarchical; instead they tend to follow more the pattern of a random graph, with nodes of many different degrees. The degree k of a node is the number of edges linking the node to neighbours. Random graphs, with arbitrary node degree distributions p_k have been studied in ref. [11], using the method of generating functionals. This method uses a continuum approximation, using derivatives to evaluate probabilities, and hence it is completely accurate only in the continuum limit of very large number of nodes N . However it can still be used to examine smaller graphs, such as those in a local area network.

We reproduce here a result from ref. [11] which give the condition for the probable existence of a giant cluster for a unipartite random graph with degree distribution p_k . (For another derivation, see Ref. [13].) This result holds in the limit $N \rightarrow \infty$. Hence in a later section we offer a correction to this large-graph result for smaller graphs.

Result 3 *The large-graph condition for the existence of a giant cluster (of infinite size) is simply*

$$\sum_k k(k-2) p_k \geq 0. \quad (16)$$

This provides a simple test that can be applied to a human-computer system, in order to estimate the possibility of complete failure via percolating damage. If we only determine the p_k , then we have an immediate machine-testable criterion for the possibility of a systemwide security breach. The condition is only slightly more complicated than the simple Cayley tree approximation; but (as we will see below) it tends to give more realistic answers.

5.3 Large random bipartite networks with arbitrary degree distributions

There is a potential pitfall associated with using Result 3 above, namely that isolated nodes (those of degree zero) are ignored. This is of course entirely appropriate if one is only concerned about percolation; but this could give a misleading impression of danger to a security administrator in the case that most of the organization shares nothing at all. In the unipartite or one-mode approximation, one-member groups with no shared files become isolated nodes; and these nodes (which are quite real to the security administrator) are not counted at all in eqn. (16). Consider for example fig. 10.

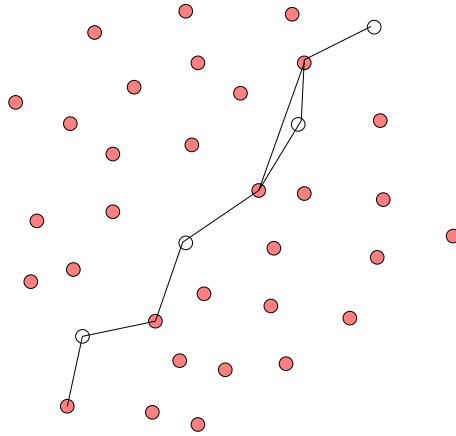


Fig. 10 A system with many isolated groups, in the unipartite representation. Is this system percolating or not? Clearly it spans a path across the diameter (order $\sqrt{N}$) of the network, but this is not sufficient to permeate an entire organization. Here we have $p_0 = 28/36$, $p_1 = 2/36$, $p_2 = 4/36$, and $p_3 = 2/36$. Result 3 tells us that the graph is percolating. Result 5 corrects this error.

It seems clear from this figure that system-wide infection is very difficult to achieve: the large number of isolated users will prevent this. However eqn. (16) ignores these users, only then considering the connected backbone—and so it tells us that this system is strongly percolating. This point is brought home even more strongly by considering fig. 11.

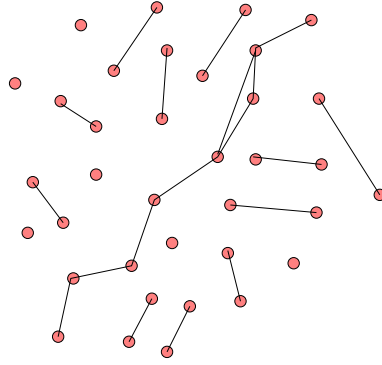


Fig. 11 The system of fig. 10, but with many more connections added. We have $p_0 = 6/36$, $p_1 = 24/36$, $p_2 = 4/36$, and $p_3 = 2/36$. Now we find the paradoxical result that adding these connections makes the system non-percolating — according to the unipartite criterion eqn. (16). Result 5 concurs.

Here we have added a fairly large number of connections. The result is that eqn. (16) now reports that the system is non-percolating. In other words, *adding* connections has caused an apparent transition from a percolating to a non-percolating state! We can avoid such problems by treating the graph in a bipartite representation. The fact which helps here is that there are no isolated nodes in this representation: files without users, and users without files, do not occur, or at least are so rare as to be of no interest to security.

Random bipartite graphs are also discussed in [11] and a corresponding expression is derived for giant clusters. Here we can let p_k be the fraction of users with degree k (ie, having access to k files), and q_k be the fraction of files to which k users have access. Then, from Ref. [11]:

Result 4 *The large-graph condition for the appearance of a giant bipartite cluster is:*

$$\sum_{jk} jk(jk - j - k) p_j q_k > 0. \quad (17)$$

This result is still relatively simple, and provides a useful guideline for avoiding the possibility of systemwide damage. That is, both distributions p_j and q_k can be controlled by policy. Thus, one can seek to prevent even the possibility of systemwide infection, by not satisfying the inequality in (17), and so holding the system below its percolation threshold.

The left hand side of (17) can be viewed as a weighted scalar product of the two vectors of degree distributions:

$$q^T W p = p^T W q > 0 \quad , \quad (18)$$

with $W_{jk} = jk(jk - j - k)$ forming a symmetric, graph-independent weighting matrix. W is infinite, but only a part of it is projected out by the non-zero degree vectors. In practice, it is sufficient to consider W up to a finite size, namely $K_{\max}$. For illustration we give W truncated at $k = 5$:

$$W_{5 \times 5} = \begin{pmatrix} -1 & -2 & -3 & -4 & -5 & \cdots \\ -2 & 0 & 6 & 16 & 30 & \\ -3 & 6 & 27 & 60 & 105 & \\ -4 & 16 & 60 & 128 & 220 & \\ -5 & 30 & 105 & 220 & 375 & \\ \vdots & & & & & \ddots \end{pmatrix} \quad (19)$$

Since W is fixed, the security administrator seeks to vary the vectors p_j and q_k so as to hold the weighted scalar product in eqn. (17) below zero, and so avoid percolation.

5.4 Application of the large-graph result for bipartite networks

Note that $W_{22} = 0$. Thus, for the case $p_2 = q_2 = 1$ (i.e. all other probabilities are zero), the expression (18) tells us immediately that any *random* graph with this degree distribution is on the threshold of percolation. We can easily find a pair of explicitly non-random graphs also having this simple degree distribution.

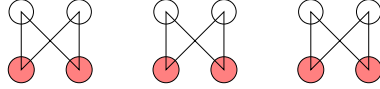


Fig. 12 A highly regular group structure which is non-percolating—but which will be pronounced ‘on threshold’ by eqn. (17).

The structures shown in figs. 12 and 13 both satisfy $p_2 = q_2 = 1$, so that the left hand side of eqn. (17) is exactly zero. One is clearly non-percolating, while the other is percolating. The typical random graph with $p_2 = q_2 = 1$ will in some sense lie ‘in between’ these two structures—hence ‘between’ percolating and non-percolating.

Now let us consider less regular structures, and seek a criterion for non-percolation. We assume that users will have many files. Hence the vector p_k will tend to have weight out to large k , and it may not be practical to try to truncate this distribution severely. Let us simply suppose that one can impose an upper limit on the number of files any user can have: $k^{\text{user}} \leq k_{\max}^{\text{user}} \equiv K$.

Now we can set up a sufficient condition for avoiding a giant cluster.

We forbid any file to have access by more than two users. That is, $q_k = 0$ for $k > 2$. This is a rather extreme constraint—it means that no group

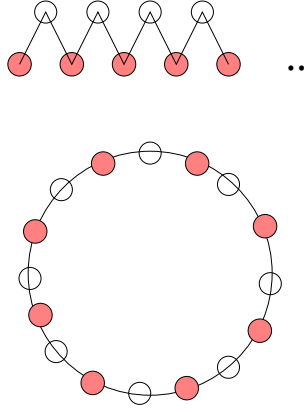


Fig. 13 Two equivalent highly regular group structures with the same ‘on-threshold’ degree distribution as in fig. 12 are the infinite chain and the ring.

has more than two members. We use this constraint here to illustrate the possible application of eqn. (17), as it is easy to obtain a sufficient condition for this case. We now demand that the first K components of Wq be negative—this is sufficient (but not necessary) to force the product $p^T Wq$ to be negative.

There are now only two nonzero q ’s, namely q_1 and q_2 . Since they are probabilities (or fractions) they sum to one. Hence, given the above truncations on p_j and q_k , we can obtain a sufficient condition for operation below the percolation threshold as

$$\begin{aligned} q_1 &\geq \frac{2K-4}{2K-3} \\ q_2 &\leq \frac{1}{2K-3}. \end{aligned} \quad (20)$$

This result seems promising; but it has a drawback which we now reveal. By letting q_1 be different from zero, we have included files with only one user. As noted in a previous section, such files contribute nothing to the connectedness of the network, and so are essentially irrelevant to security concerns. In fact, q_1 is simply the fraction of private files; and it is clear from our solution that, if we want our users to have many files (large K), then nearly all of them must be private ($q_1 \rightarrow 1$ as $K \rightarrow \infty$). This in turn means that the group structure is rather trivial: there are a very few groups with two members (those files contributing to q_2), and the rest of the groups are one-member groups with private files.

We can find other solutions by explicitly setting $q_1 = 0$: that is, we do not count the private files. If we retain the truncation rules above ($k_{\max} = K$ for users, and $k_{\max} = 2$ for files), then we get $q_2 = 1$ immediately. This

allows us to collapse the sum in eqn. (17) to $2 \sum_k k(k-2) p_k$. This gives the same threshold criterion as that for a unipartite graph [eqn. (16)]. However this is only to be expected, since with $q_2 = 1$, the bipartite structure is trivial: it amounts to taking a unipartite graph (of users) and placing a file on each inter-user link. The group structure is equally trivial: every group is elementary and has two users, and all connections between groups are mediated by shared users. This situation also seems ill-suited to practical computer systems operation.

Finally, we hold $q_1 = 0$ but allow both q_2 and q_3 to be nonzero. We then find that our simple strategy of forcing the first K components of Wq to be zero is impossible. And of course allowing even larger degree for the files does not help. Hence, for this, most practical, case, some more sophisticated strategy must be found in order to usefully exploit the result (17).

5.5 Small-graph corrections

The problem with Results 3 and 4 is that they are derived under the assumption that the number of nodes is diverging. For a small graph with N nodes (either unipartite or bipartite), the criterion for a giant cluster becomes inaccurate. Clusters do not grow to infinity, they can only grow to size N at the most, hence we must be more precise and use a finite scale rather than infinity as a reference point.

A more precise percolation criterion states that, at percolation, the average size of clusters is of the same order of magnitude as the number of nodes. However for a small graph the size of a giant cluster and of below-threshold clusters [N and $\log(N)$, respectively] are not that different[14].

For a unipartite graph we can state [13] the criterion for percolation more precisely as:

$$\frac{\langle k \rangle^2}{-\sum_k k(k-2) p_k} \gg 1. \quad (21)$$

(Result 3 then follows by requiring the denominator to go to zero, so forcing the quantity in Eq. 21 to diverge.) Then [13] a finite-graph criterion for the percolation threshold can be taken to be as follows.

Result 5 *The small-graph condition for widespread percolation in a unipartite graph of order N is:*

$$\langle k \rangle^2 + \sum_k k(k-2) p_k > \log(N). \quad (22)$$

This can be understood as follows. If a graph contains a giant component, it is of order N , and the size of the next largest component is typically $O(\log N)$; thus, according to the theory of random graphs the margin for error in estimating a giant component is of order $\pm \log N$. Eq. 21, giving the criterion for a cluster that is much greater than unity, implies that the right

hand side of Eq. 22 is greater than zero. To this we now add the magnitude ($\log(N)$) of the uncertainty, in order to reduce the likelihood of an incorrect conclusion.

Similarly, for the bipartite graph, one has

Result 6 *The small-graph condition for widespread percolation in a bipartite graph of order N is:*

$$\langle k \rangle^2 \langle j \rangle^2 + \sum_{jk} jk(jk - j - k) p_j p_k > \log(N/2). \quad (23)$$

These expressions are not much more complex than the large-graph criteria. Moreover, they remain true in the limit of large N . Hence we expect these small-graph criteria to be more reliable than the large-graph criteria for testing percolation in small systems. This expectation is borne out in the examples below. In particular, we find that, since the average coordination number $\langle k \rangle$ enters into the small-graph percolation criteria, the earlier problem of ignoring isolated nodes in the unipartite case is now largely remedied.

Thus we expect that Results 5 and 6 should give a more accurate guide to the system administrator who seeks to prevent the spreading of harmful information, by preventing the system from forming a single connected cluster (ie, from percolating).

5.6 Examples

We have obtained several distinct criteria for finding the limits at which graphs start to percolate. These criteria are based on statistical, rather than exact, properties of the graph to be tested. They thus have the advantage of being applicable even in the case of partial information about a graph; and the disadvantage that they are approximate. The principal approximation in all of them stems from the assumption that graphs with a given p_k distribution are typical of a random ensemble of such graphs. Other approximations are also present (the CT assumptions; and/or the assumption of large N).

For small, fixed graphs there is sometimes no problem in exploring the whole graph structure and then using an exact result. How then do the above limits compare with more simplistic (but nonstatistical) criteria for graph percolation? For example, one could—naively—simply compare the number of links in a graph to the minimum number required to connect all of the nodes:

$$L > N - 1. \quad (24)$$

This is a necessary but not sufficient criterion for percolation; hence, when it errs, it will give false positives. This condition holds as an equality for an open chain, and also for the 'star' or 'hub' topology.

Perhaps the most precise small-graph criterion for percolation comes from asking how many pairs of nodes, out of all possible pairs, can reach one another in a finite number of hops. We thus define the ratio R_C of connected pairs of nodes to the total number of pairs that could be connected:

$$R_C = \sum_{i=\text{clusters}} \frac{\frac{1}{2}n_i(n_i - 1)}{\frac{1}{2}N(N - 1)} = 1. \quad (25)$$

This is simply the criterion that the graph be connected.

If we wish to simplify this rule for ease of calculation, we can take $n_i \approx L_i + 1$, where L_i is the number of links in cluster i . Then, if L is the total number of links in the graph, criterion (25) becomes

$$R_L = \frac{L(L + 1)}{N(N - 1)} > 1. \quad (26)$$

This latter form is in fact the same as (24). Thus we are left with one 'naive' small-graph test which is very simple, and one 'exact' criterion which requires a little more work to compute.

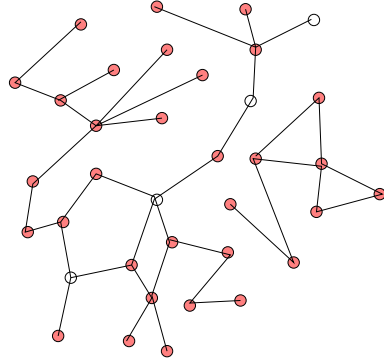


Fig. 14 The same nodes as figs. 10 and 11, but now with almost complete connectivity. $p_0 = 0$, $p_1 = 13/36$, $p_2 = 12/36$, $p_3 = 6/36$, $p_4 = 4/36$, $p_5 = 1/36$.

We now apply the various tests—both the statistical tests, and those based on exact knowledge of the graph—to several of the graphs shown in this paper. Table 1 summarizes the results. (The bipartite criterion (17) is not included in the table, as it cannot be used with all the examples.) It should be noted that none of the criteria are flawless in their estimation of percolation, except for the exact count of pairs. One positive aspect of the errors with the approximate tests is that they are all false positives—ie, they warn of percolation even though the graph is not fully percolating. False positives are presumably more tolerable for security concerns than false negatives.

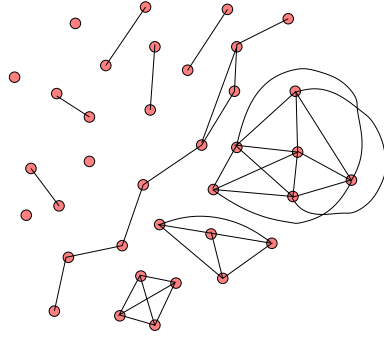


Fig. 15 The system of fig. 11, but with high densities of local nodes added that artificially increase the link density without improving global connectivity significantly. This type of graph fools most of the criteria.

We also (last column) rate the example graphs according to how many of the criteria are correct for each graph. Here we see that it is possible to find link configurations that fool all of the approximate tests into signalling percolation where, in fact, there is none. That is, Figs. 14 and 15 give a lot of false positives.

Let us try to summarize the properties of these various percolation tests. First, it seems from our limited sample that the various (nonexact) tests are roughly comparable in their ability to detect percolation. Also, they all need roughly the same information—the node degree distribution p_k —except for Eqn. (24). Hence (24) may be useful as a very inexpensive first diagnostic, to be followed up by others in case of a positive result. The statistical tests we have examined are useful when only partial information about a graph is available. One can even use partial information to estimate the parameters in (24), as well as to determine (based on the test) whether further information is desirable. Of the statistical tests, the Eqn. (16) appears to be the least reliable.

The exact test (25) is easy to compute if one has the entire graph: it can be obtained using a depth-first search, requiring time of $O(N + L)$ —the same amount of time as is needed to obtain the graph. Also, it gives a good measure of closeness to the threshold for nonpercolating graphs (useful, for example, in the case of figure 14).

Finally we offer more detailed comments.

1. Note that, for fig. 10, the large (unipartite) graph criterion in eqn. (16) claims that the graph has a giant cluster, which is quite misleading. The bipartite test (17) cannot be relevant here, since we must then view the isolated nodes as either users without files, or files without users—either event being improbable for a user/file system. However, the small graph criterion in eqn. (22) gives the correct answer.
2. In fig. 11, we are still able to fool the large graph criterion, in the sense that (as noted earlier) the large graph criterion in eqn. (16) claims that

Fig.	Perc.	Eqn. 15	Eqn. 16	Eqn. 22	Eqn. 24	Eqn. 25	Score
6	yes	$2.33 \geq 0.83$	$1.87 \geq 0$	$6.42 \geq 2.71$	$16 \geq 14$	$1 = 1$	5/5
10	no	$0.44 \geq 1.5$	$0.11 \geq 0$	$0.31 \geq 3.58$	$8 \geq 35$	$0.04 = 1$	4/5
11	no	$1.06 \geq 1.5$	$-0.5 \geq 0$	$0.61 \geq 3.58$	$19 \geq 35$	$0.06 = 1$	5/5
14	no	$2.11 \geq 1.25$	$1.44 \geq 0$	$5.90 \geq 3.58$	$38 \geq 35$	$0.67 = 1$	1/5
15	no	$2.17 \geq 1.25$	$2.61 \geq 0$	$7.31 \geq 3.58$	$39 \geq 35$	$0.095 = 1$	1/5
Hub	yes	$1.94 \geq 1.03$	$31.11 \geq 0$	$34.89 \geq 3.58$	$35 \geq 35$	$1 = 1$	5/5
Score	-	4/6	3/6	4/6	4/6	6/6	-

Table 1 Table summarizing the correctness of the tests for percolation through small graphs and their reliability scores. The final entry ‘Hub’ is for comparison and refers to a graph of equal size to figures 10-15 in which a single node connects all the others by a single link in hub configuration.

fig. 11 is safer than fig. 10—even though the former is clearly more connected. Again the small-graph test gives us the right answer.

3. In fig. 14, we find the same graph, now almost completely connected. In fact, a single link suffices to make the graph connected. For this highly connected case, all the tests except the exact one are fooled—perhaps a case of acceptable false positives. Fig. 15—which is much farther from percolation—also fools all the tests. And it is plausible that many real user/file systems resemble fig. 15, which thus represents perhaps the most strong argument for seeking complete information and using the exact test for percolation.
4. It appears that none of our tests give false negatives. Further evidence comes from testing all of the above percolation criteria using Fig. 1. Here all the tests report percolation—again, no false negatives.
5. We note again that criterion (16) can be misleading whenever there are many isolated users. For example, consider adding $N_i - 1$ isolated users (but with files) to fig. 3b (so that the number of isolated users is N_i). The unipartite criterion will ignore these, and report that the graph is percolating, for any N_i . The bipartite criterion (17) gives a completely different answer: it gives a non-percolating structure for any physical N_i (i.e. for any $N_i \geq 0$). The unipartite small-graph criterion—expected to be more accurate than (16)—gives, for $N_i = 1$ (the case in the figure), $5.7 > 2.7$ —the graph is identified as percolating.
6. Finally we can consider applying eqn. (16) to shorthand graphs, as they are considerably more compact than the full microscopic graph. For the shorthand graph one can treat elementary groups as nodes, and dashed and solid lines as links. For large systems it may be useful to work with such graphs—since the groups themselves may be defined for practical (collaborative) reasons and so cannot be altered—but connections between the groups can be. Thus it may be useful to study the percolation approximation for such shorthand graphs. However, the shorthand graphs for the above figures are too small for such an exercise to be meaningful.

6 Node centrality and the spread of information

In the foregoing sections, we have considered groups of users as being either unconnected, or else only weakly bound, below or near the percolation threshold. Although there may be circumstances in which this situation is realistic, there will be others in which it is not: collaborative needs, as well as social habits, often push the network towards full connectivity. In this case, one infected node could infect all the others. (As noted in Section 2, an example of this occurred recently at a Norwegian University, where the practice of using a single administrator password on all hosts, allowed a keyboard sniffer to open a pathway that led to all machines, through this common binding. There was effectively a giant hub connection between every host, with a single node that was highly exposed at the centre.)

In this section, we consider the *connected* components of networks and propose criteria for deciding which nodes are most likely to infect many other nodes, if they are compromised. We do this by examining the relative connectivity of the graph along multiple pathways.

What are the best connected nodes in a graph? These are certainly nodes that an attacker would like to identify, since they would lead to the greatest possible access, or spread of damage. Similarly, the security auditor would like to identify them and secure them, as far as possible. From the standpoint of security, then, important nodes in a network (files, users, or groups in the shorthand graph) are those that are 'well-connected'. Therefore we seek a precise working definition of 'well-connected', in order to use the idea as a tool for pin-pointing nodes of high security risk.

A simple starting definition of well-connected could be 'of high degree': that is, count the neighbours. We want however to embellish this simple definition in a way that looks beyond just nearest neighbours. To do this, we borrow an old idea from both common folklore and social network theory[15]: an important person is not just well endowed with connections, but is well endowed with connections to important persons.

The motivation for this definition is clear from the example in figure 16. It is clear from this figure that a definition of 'well-connected' that is relevant to the diffusion of information (harmful or otherwise) must look beyond first neighbours. In fact, we believe that the circular definition given above (important nodes have many important neighbours) is the best starting point for research on damage diffusion on networks.

Now we make this verbal (circular) definition precise. Let v_i denote a vector for the importance ranking, or connectedness, of each node i . Then, the importance of node i is proportional to the sum of the importances of all of i 's nearest neighbours:

$$v_i \propto \sum_{j=\text{neighbours of } i} v_j \quad . \quad (27)$$

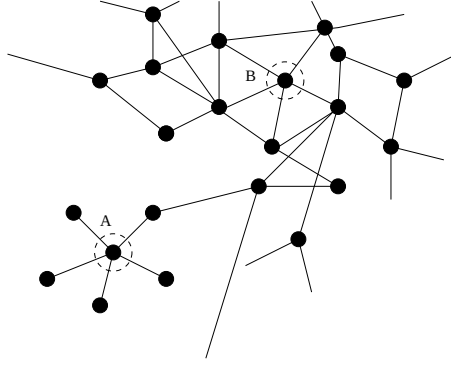


Fig. 16 Nodes *A* and *B* are both connected by five links to the rest of the graph, but node *B* is clearly more important to security because its neighbours are also well connected.

This may be more compactly written as

$$v_i = (\text{const}) \times \sum_j A_{ij} v_j , \quad (28)$$

where *A* is the *adjacency matrix*, whose entries A_{ij} are 1 if *i* is a neighbour of *j*, and 0 otherwise. We can rewrite eqn. (28) as

$$A \mathbf{v} = \lambda \mathbf{v} . \quad (29)$$

Now one sees that the importance vector is actually an eigenvector of the adjacency matrix *A*. If *A* is an $N \times N$ matrix, it has *N* eigenvectors (one for each node in the network), and correspondingly many eigenvalues. The eigenvalue of interest is the principal eigenvector, i.e. that with highest eigenvalue, since this is the only one that results from summing all of the possible pathways with a positive sign. The components of the principal eigenvector rank how self-consistently ‘central’ a node is in the graph. Note that only ratios v_i/v_j of the components are meaningfully determined. This is because the lengths $v^i v_i$ of the eigenvectors are not determined by the eigenvector equation.

The highest valued component is the most central, i.e. is the *eigencentre* of the graph. This form of well-connectedness is termed ‘eigenvector centrality’ [15] in the field of social network analysis, where several other definitions of centrality exist. For the remainder of the paper, we use the terms ‘centrality’ and ‘eigenvector centrality’ interchangeably.

We believe that nodes with high eigenvector centrality play a correspondingly important role in the diffusion of information on a network. However, we know of only few previous studies (see ref. [16]) which test this idea quantitatively. We propose that this measure of centrality should be adopted as a diagnostic instrument for identifying the best connected nodes in networks of users and files.

To illustrate our idea we calculate the centrality for the nodes in some of the figures. In fig. 1, simple neighbour counting suggests that the file 'b' is most well-connected. However the node with highest centrality is actually user 5 (by a small margin of about 1%). After the fact, one can see reasons for this ranking by studying the network. The point, however, is already clear from this simple example: the rankings of connectedness can change in non-trivial ways if one moves from simple degree counting to the self-consistent definition of centrality. The centrality component-values for fig. 1 are ranked as follows:

$$\begin{aligned} 5 \approx b > c = d \approx 3 = 4 \\ > e > 2 > 6 = 7 \approx a > 1 . \end{aligned} \quad (30)$$

This ranking could be quite useful for a system administrator concerned with security: it allows the direct comparison of security of files and users on a common footing.

For large graphs, visual inspection fails to give sufficient insight, while the calculation of the centrality is trivial. Graphs tend to be sparse as they grow large, which makes diagonalization of the adjacency matrix quick and easy. For fig. 6, we find

$$\begin{aligned} 1 > 2 >> B = D > G > F > 4 \\ > A = C = E > 3 > I > J = K > H . \end{aligned} \quad (31)$$

Again we see results that are not so intuitive. File 4 is ranked significantly lower than file 2, even though it has the same degree. Our intuition is not good for comparing the files to the users in this bipartite graph; nor does it help much in comparing the users to one another, as they all have degree one or two. The point, once again, is that it is an effect that goes beyond nearest neighbours that matters. Computing the centrality for fig. 7, the one-mode version of fig. 6, gives

$$\begin{aligned} B = D > G > F > A = C \\ = E > I > J = K > H . \end{aligned} \quad (32)$$

Note that the inter-user rankings have not changed. We conjecture that this is a general property of the centrality under the one-mode projection, contingent on weighting the effective inter-user links in the projection with the number of files they represent. Whether or not this conjecture is rigorously true, the projected graph may be useful in some circumstances, when it is inconvenient to track the population of files.

We note also that it is instructive to compare the ranking in eqn. (32) with the Venn diagram in fig. 8. We suggest that the Venn diagram may be the most transparent pictorial rendition of connectedness (as measured by eigenvector centrality) that we have examined here; i.e. the Venn diagram most clearly suggests the ranking found by eigenvalue centrality in eqn. (32).

Next we look at the shorthand graph (Fig. 9) and choose to retain the information on the number of links (which can be greater than one in the shorthand graph) in the shorthand A matrix. The entries now become non-negative integers[16]. With this choice, we obtain these results: the central node has highest centrality (0.665), followed by the leftmost node (0.5), and finally by the two rightmost nodes (0.4). The importance of the central node is obvious, while the relative importance of the right and left sides is less so. Of course, larger shorthand graphs will still confound intuition, so the analysis may also be a useful tool in analyzing security hot-spots at the shorthand, group level.

The above examples illustrate some of the properties of eigenvector centrality on small graphs. Now we wish to consider the application of the same concept to larger graphs. Here there is a crucial difference from the percolation tests considered in the previous section. That is, testing for percolation simply gives a single answer: Yes, No, or somewhere in between. However a calculation of node centrality gives a score or index for every node on the network. This is a large amount of information for a large graph; the challenge is then to render this information in useful form.

Here we present an approach to meeting this challenge; for details see [17]. When a node has high eigenvector centrality (EVC), it *and its neighbourhood* have high connectivity. Thus in an important sense EVC scores represent neighbourhoods as much as individual nodes. We then want to use these scores to define clusterings of nodes, with as little arbitrariness as possible. (Note that these clusterings are not the same as user groups—although groups are unlikely to be split up by our clustering approach.)

To do this, we define as Centres those nodes whose EVC is higher than any of their neighbours' scores. Clearly these Centres are important in the flow of information on the network. We also associate a Region (subset of nodes) with each Centre. These Regions are the clusters that we seek. We find (see [17]) that more than one rule may be reasonably defined to assign nodes to Regions; the results differ in detail, but not qualitatively. One simple rule is to use distance (in hops) as the criterion: a node belongs to a given Centre (ie, to its Region) if it is closest (in number of hops) to that Centre. With this rule, some nodes will belong to multiple regions, as they are equidistant from two or more Centres. This set of nodes defines the Border set.

The picture we get then is of one or several regions of the graph which are well-connected clusters—as signalled by their including a local maximum of the EVC. The Border then defines the boundaries between these regions. This procedure thus offers a way of coarse-graining a large graph. This procedure is distinct from that used to obtain the shorthand graph; the two types of coarse-graining may be used separately, or in combination.

To illustrate these ideas we present data from a large graph, namely, the Gnutella peer-to-peer file-sharing network, viewed in a snapshot taken November 13, 2001 [18]. In this snapshot the graph has two disconnected pieces—one with 992 nodes, and one with three nodes. We ignore the small

piece (the graph, viewed as a whole, is 99.4% percolating according to Eq. (25)), and analyze the large one. Here we find that the Gnutella graph is very well-connected. There are only two Centres, hence only two natural clusters. These regions are roughly the same size (about 200 nodes each). This means, in turn, that there are many nodes (over 550!) in the Border.

In figure 17 we present a visualization of this graph, using Centres, Regions, and the Border as a way of organizing the placement of the nodes using our Archipelago tool[19].

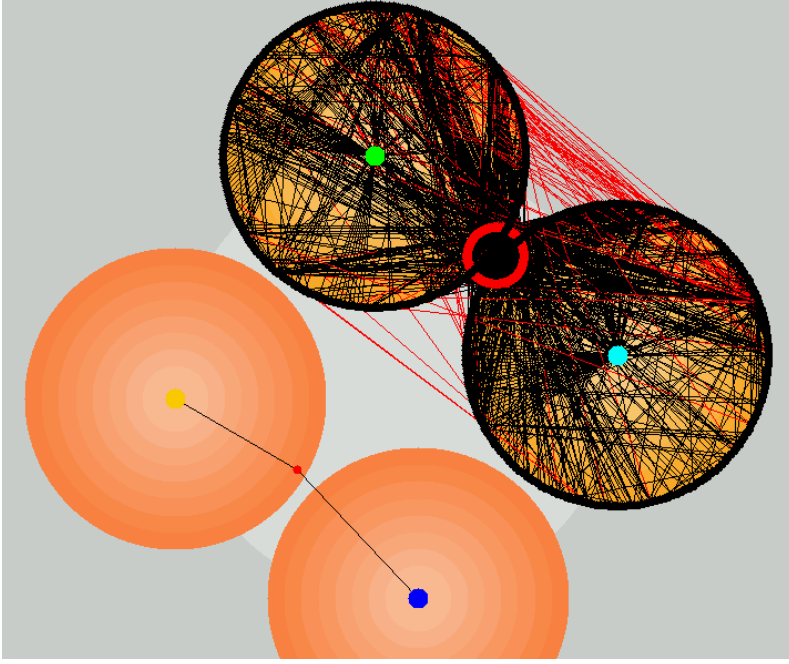


Fig. 17 A top level, simplified representation of Gnutella peer to peer associations, organized around the largest centrality maxima. The graph consists of two fragments, one with 992 nodes and one of merely 3 nodes and organizes the graph into Regions. The upper connected fragment shows two regions connected by a ring of bridge nodes.

Both the figure and the numerical results support our description of this graph as well-connected: it has only a small number of Regions, and there are many connections (both Border nodes, and links) between the Regions. We find these qualitative conclusions to hold for other Gnutella graphs that we have examined. Our criteria for a well-connected graph are consonant with another one, namely, that the graph has a power-law node degree distribution [9]. Power-law graphs are known to be well-connected in the sense that they remain connected even after the random removal of

a significant fraction of the nodes. And in fact the (self-organized) Gnutella graph shown in fig. 17 has a power-law node degree distribution.

We believe that poorly-connected (but still percolating) graphs will be revealed, by our clustering approach, to have relatively many Centres and hence Regions, with relatively few nodes and links connecting these Regions. Thus, we believe that the calculation of eigenvector centrality, followed by the simple clustering analysis described here (and in more detail in [17]), can give highly useful information about how well connected a graph is, which regions naturally lie together (and hence allow rapid spread of damaging information), and where are the boundaries between such easily-infected regions. All of this information should be of utility in analyzing a network from the point of view of security.

7 Conclusions

We have constructed a graphical representation of security and vulnerability within a human-computer community, using the idea of association and connectedness. We have found a number of analytical tests that can be applied to such a representation in order to measure its security, and find the pivotal points of the network, including those that are far from obvious. These tests determine approximately when a threshold of free flow of information is reached, and localize the important nodes that underpin such flows.

We take care to note that the measurements we cite here depend crucially on where one chooses to place the boundaries of one's network analysis. The methods will naturally work best, when no artificial limits are placed on communication, e.g. by restricting to a local area network if there is frequent communication with the world beyond its gateway. On the other hand, if communication is dominated by local activity (e.g. by the presence of a firewall) then the analysis can be successfully applied to a smaller vessel.

A number of different approaches were considered in this paper, all of which can now be combined into a single model. The results of section 4 are equivalent to adding an appropriate number of additional end nodes (degree one) as satellites to a hub at every node where an external exposure is non-zero. This internalizes the external bath and weights the node's relative connectivity and acts as an effective embedding into a danger bath. This alters the centrality appropriately and also adds more end nodes for percolation routes. Thus we can combine the information theoretical considerations with the graphical ones entirely within a single representation.

At the start of this paper, we posed some basic questions that we can now answer.

1. *Is there an optimal number of file and user groups and sizes?* Here the answer is negative. There is actually an answer which optimizes security—namely, let $G = N$: all groups are one-user groups, completely isolated from one another. This answer is however useless, as it ignores

all the reasons for having connectivity among users. Hence we reject this "maximum-security, minimum-communication" optimum. Then it is not the number of collaborations that counts, but rather how much they overlap and how exposed they are. The number of files and users in a group can increase its exposure to attack or leakage, but that does not follow automatically. Risk can always be spread. The total number of points of contact is the criterion for risk, as intuition suggests. Information theory tells us, in addition, that nodes that are exposed to potential danger are exponentially more important, and therefore require either more supervision, or should have exponentially fewer contact points.

2. *How do we identify weak spots?* Eigenvalue centrality is the most revealing way of finding a system's vulnerable points. In order to find the true eigencentre of a system, one must be careful to include every kind of association between users, i.e. every channel of communication, in order to find the true centre.
3. *How does one determine when system security is in danger of breaking down?* We have provided two simple tests that can be applied to graphical representations (results 5 and 6). These tests reveal what the eye cannot necessarily see in a complex system, namely when its level of random connectivity is so great that information can percolate to almost any user by some route. These tests can easily be calculated. The appearance or existence of a giant cluster is not related to the number of groups, but rather to how they are interconnected.

An attacker could easily perform the same analyses as a security administrator and, with only a superficial knowledge of the system, still manage to find the weak points. An attacker might choose to attack a node that is close to a central hub, since this attracts less attention but has a high probability of total penetration, so knowing where these points are allows one to implement a suitable protection policy. It is clear that the degree of danger is a policy dependent issue: the level of acceptable risk is different for each organization. What we have found here is a way of comparing strategies, that would allow us to minimize the relative risk, regardless of policy. This could be used in a game-theoretical analysis as suggested in ref. [20]. The measurement scales we have obtained can easily be programmed into an analysis tool that administrators and security experts can use as a problem solving 'spreadsheet' for security. We are constructing such a graphical tool that administrators can use to make informed decisions[19].

There are many avenues for future research here. We have ignored the role of administrative cost in maintenance of groups of different size. If groups change quickly the likelihood of errors in membership increases. This can lead to the formation of ad hoc links in the graph that provide windows of opportunity for the leakage of unauthorized information. Understanding the percolation behaviour in large graphs is a major field of research; several issues need to be understood here, but the main issue is how a graph splits into different clustersIEEE Computer Society Press in real computer systems. There are usually two mechanisms at work in social graphs: purely

random noise and a ‘rich get richer’ accumulation of links at heavily connected sites. Further ways of measuring centrality are also being developed and might lead to new insights. We have ignored the directed nature of the graph in this work, for the purpose of clarity; this is a detail that we do not expect to affect our results except in minor details. We hope to return to some of these issues in future work.

Acknowledgement: We are grateful to Jonathan Laventhol for first getting us interested in the problem of groups and their number. GC and KE were partially supported by the Future & Emerging Technologies unit of the European Commission through Project BISON (IST-2001-38923).

References

1. L. Snyder. Formal models of capability-based protection systems. *IEEE Transactions on Computers*, 30:172, 1981.
2. L.E. Moser. Graph homomorphisms and the design of secure computer systems. *Proceedings of the Symposium on Security and Privacy, IEEE Computer Society*, page 89, 1987.
3. J.C. Williams. A graph theoretic formulation of multilevel secure distributed systems: An overview. *Proceedings of the Symposium on Security and Privacy, IEEE Computer Society*, page 97, 1987.
4. M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. *Computer Communications Review*, 29:251, 1999.
5. A.L. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: topology of the world-wide web. *Physica A*, 281:69, 2000.
6. A.L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509, 1999.
7. R. Albert, H. Jeong, and A.L. Barabási. Diameter of the world-wide web. *Nature*, 401:130, 1999.
8. B. Huberman and A. Adamic. Growth dynamics of the world-wide web. *Nature*, 401:131, 1999.
9. R. Albert and A. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74:47, 2002.
10. M.Y. Kao. Data security equals graph connectivity. *SIAM Journal on Discrete Mathematics*, 9:87, 1996.
11. M. E. J. Newman, S.H. Strogatz, and D.J. Watts. Random graphs with arbitrary degree distributions and their applications. *Physical Review E*, 64:026118, 2001.
12. D. Brewer and M. Nash. The chinese wall security policy. In *Proceedings of the IEEE Symposium on Security and Privacy*, page 206. IEEE Computer Society Press, 1989.
13. M. Burgess. *Analytical Network and System Administration — Managing Human-Computer Systems*. J. Wiley & Sons, Chichester, 2004.
14. M. Molloy and B. Reed. The size of the giant component of a random graph with a given degree sequence. *Combinatorics, Probability and Computing*, 7:295, 1998.
15. P. Bonacich. Power and centrality: a family of measures. *American Journal of Sociology*, 92:1170–1182, 1987.

16. G. Canright and Å. Weltzien. Multiplex structure of the communications network in a small working group. *Proceedings, International Sunbelt Social Network Conference XXIII, Cancun, Mexico*, 2003.
17. G. Canright and K. Engø-Monsen. A natural definition of clusters and roles in undirected graphs. *Science of Computer Programming*, (in press), 2004.
18. Mihajlo Jovanovic. *Private communication*, 2001.
19. M. Burgess, G. Canright, T. Hassel Stang, F. Pourbayat, K. Engo, and Å. Weltzien. Archipelago: A network security analysis tool. *Proceedings of the Seventeenth Systems Administration Conference (LISA XVII) (USENIX Association: Berkeley, CA)*, page 153, 2003.
20. M. Burgess. Theoretical system administration. *Proceedings of the Fourteenth Systems Administration Conference (LISA XIV) (USENIX Association: Berkeley, CA)*, page 1, 2000.