



## The Secur//Tree<sup>®</sup> BurgleHouse Tutorial (a.k.a., Who wants to be a Cat Burglar?)

### Introduction

Secur//Tree<sup>®</sup> is a state of the art, *attack tree*-based risk assessment tool. It revolutionizes how we view threats against our assets. Although it is easy to use, the concepts upon which it is based are unfamiliar to many people. Through this sample exercise you will not only become familiar with the use of the tool, but also with the main concepts of *capabilities-based attack tree analysis*. Most people find it takes about one hour to complete the exercise.

Secur//Tree models how assets or systems of almost any kind can be attacked. Its greatest use to date has been in the field of Information Technology (IT). Increasingly, however, people are using Secur//Tree to analyze threats against other assets ranging from aircraft avionics to gas pipelines and electrical transmission lines. To make it possible for all readers to appreciate the power of Secur//Tree, this tutorial chooses a situation of concern to almost everyone – a residential house burglary. Although the example is simplistic, it demonstrates that attack tree analysis is not limited to IT situations. In fact, one of the novel things about Secur//Tree is its ability to model the physical threats to information systems and electronic threats to physical systems.

### Attack Tree Modeling

Secur//Tree uses a graphical construct known as an *attack tree*<sup>1</sup> to model attacks. The attack tree model describes the vulnerabilities in a system and the actions that could be performed by an adversary in order for an attack to be successful. The model includes estimates of the resources needed to carry out specific attacks, the impact of those attacks on the victim and the benefits realized by the attacker.

---

<sup>1</sup> *Attack trees* originate from work done by a number of researchers including:

J.D. Weiss, A System Security Engineering Process, Proceedings of the 14<sup>th</sup> National Computer Security Conference, 1991.

Edward G. Amoroso, Fundamentals of Computer Security Technology, pp 15-29, Prentice-Hall, 1994 ISBN0131089293

B. Schneier, Seminar session given at a Computer Security Institute conference in November, 1997.

B. Schneier, Attack Trees: Modeling Actual Threats, SANS Network Security 99 – The Fifth Annual Conference on UNIX and NT Network Security, New Orleans, Louisiana. Wednesday, October 6<sup>th</sup>, 1999, Session Two, Track One - Invited Talks

B. Schneier, Attack Trees, Dr. Dobb's Journal, v. 24, n. 12, December 1999, pp. 21-29.

Moore, A., Ellison, R. and R. Linger, "Attack Modeling for Information Security and Survivability", March 2001, <http://www.cert.org/archive/pdf/01tn001.pdf>

Shelby Evans, David Heinbuch, Elizabeth Kyle, John Piorkowski, James Wallner, "Risk-Based Systems Security Engineering: Stopping Attacks with Intention", November/December 2004, IEEE Security and Privacy



# Amenaza

TECHNOLOGIES LIMITED

There are many ways to attack almost any system but very few organizations have the time and resources to protect against every eventuality. Therefore, it is essential that we find a way to predict which attacks are most likely and make strategic decisions about our defenses.

Secur//Tree is first used to construct a model of the system and its vulnerabilities. Then, the risk analyst identifies the classes of individuals who have a motive for harming the system being modeled. These classes of hostile individuals are known as *threat agents*. Different *threat agents* have different characteristics and capabilities. By comparing the resources needed to perform an attack to the capabilities of a *threat agent*, Secur//Tree predicts how and where an attack will occur. **Identifying the likely avenues of attack makes it cost effective to construct workable defenses.**

This tutorial illustrates the attack tree analytic process by examining the activities that could result in the burglary of a typical residential dwelling. Using attack tree analysis, we will determine how two types of burglars would be most likely to accomplish this fiendish feat.

## Software Requirements

This quick tour assumes you have already installed a current Java™ Virtual Machine (JVM) (from <http://www.java.com>) and the Secur//Tree application (from a Secur//Tree evaluation CD or from the Amenaza website, <http://www.amenaza.com>). Secur//Tree runs on most major computing platforms including Windows™ (Window NT, Windows 2000, Windows XP), Macintosh™ and Linux. In order to activate the Secur//Tree software you must obtain a license key file from Amenaza Technologies Limited and copy it to the Secur//Tree installation directory. If you need help with any of these steps, please contact Amenaza at 888-949-9797 (toll-free in US and Canada), or +01-403-263-7737 (international), or via email ([support@amenaza.com](mailto:support@amenaza.com)).

Secur//Tree data files end in the extension *.rit* (which stands for RIsk Tree). In the same way that a word processor (such as Microsoft Word™) opens document files (which end in *.doc*) Secur//Tree works with files ending in *.rit*.


We have created a special data file for you to work on as you follow this guide. The file is called, *BurgleHouse.rit*. The default installation of Secur//Tree places it in `C:\Program Files\Amenaza\SecurITree\Data`. If you cannot locate *BurgleHouse.rit* please contact Amenaza at one of the numbers listed previously. Since you will be editing *BurgleHouse.rit* you might want to make a copy of it before you start the tutorial. That way, if you make a mistake, or simply want to experiment, it will be easy start over.

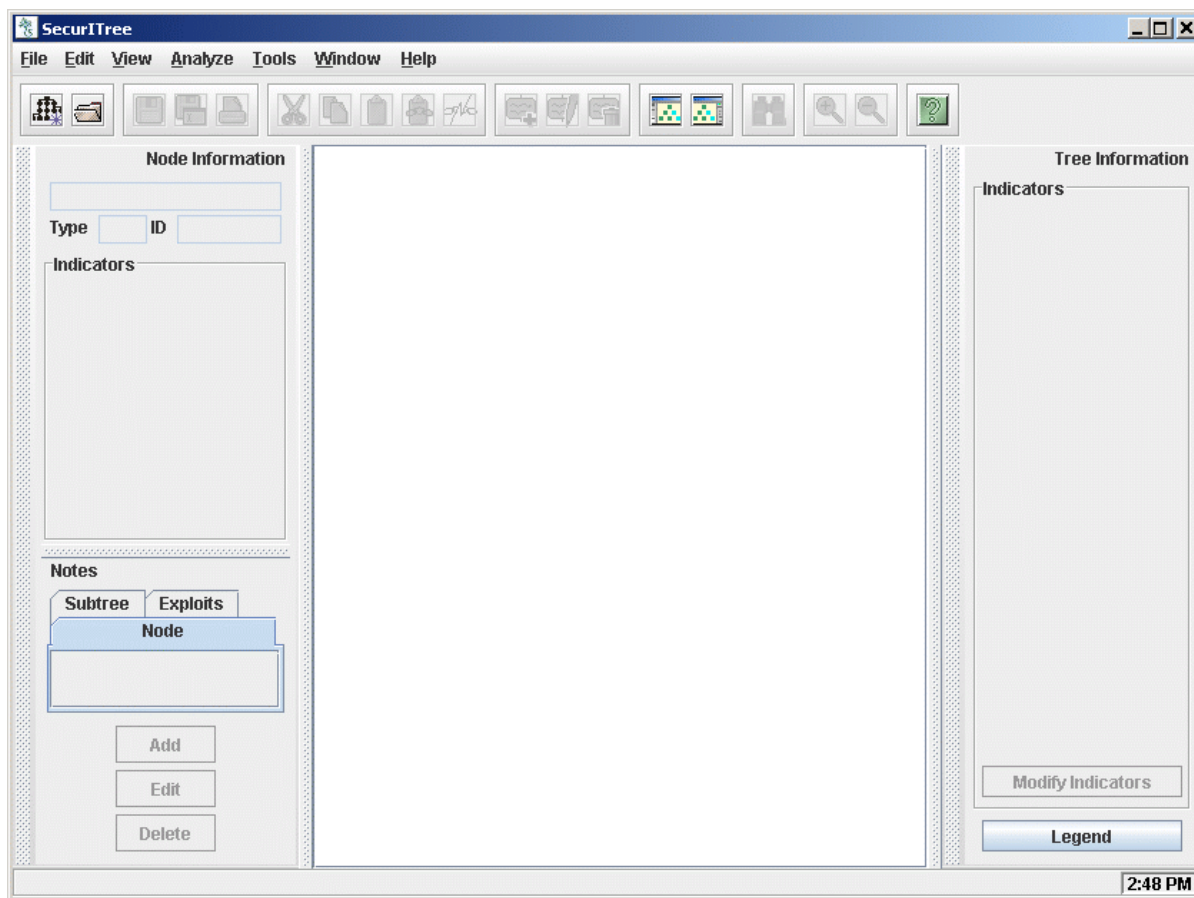
The demonstration is not meant to be comprehensive. There are many features of the model and tool that are not discussed in this document.





## The Tour

### Capabilities-based Attack Tree Analysis

When Secur//ree<sup>®</sup> was installed an icon  should have been created on your desktop or under the *Start* menu used by your operating system. On a Windows<sup>™</sup> platform this will typically be found under *Start > Programs > Amenaza SecurITree > Amenaza SecurITree* (or simply double-click on the icon on your desktop). After a few moments (about 15 seconds), Secur//ree will launch. The following window will appear (see **Figure 1**).<sup>2</sup>

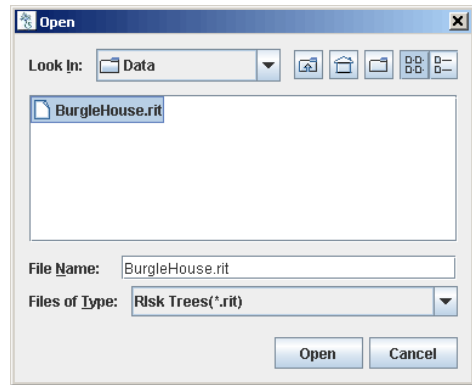


**Figure 1**

<sup>2</sup> If either of the side panels (labeled *Node Information* and *Tree Information*) do not appear, enable them by clicking on the toolbar buttons (  or  ).

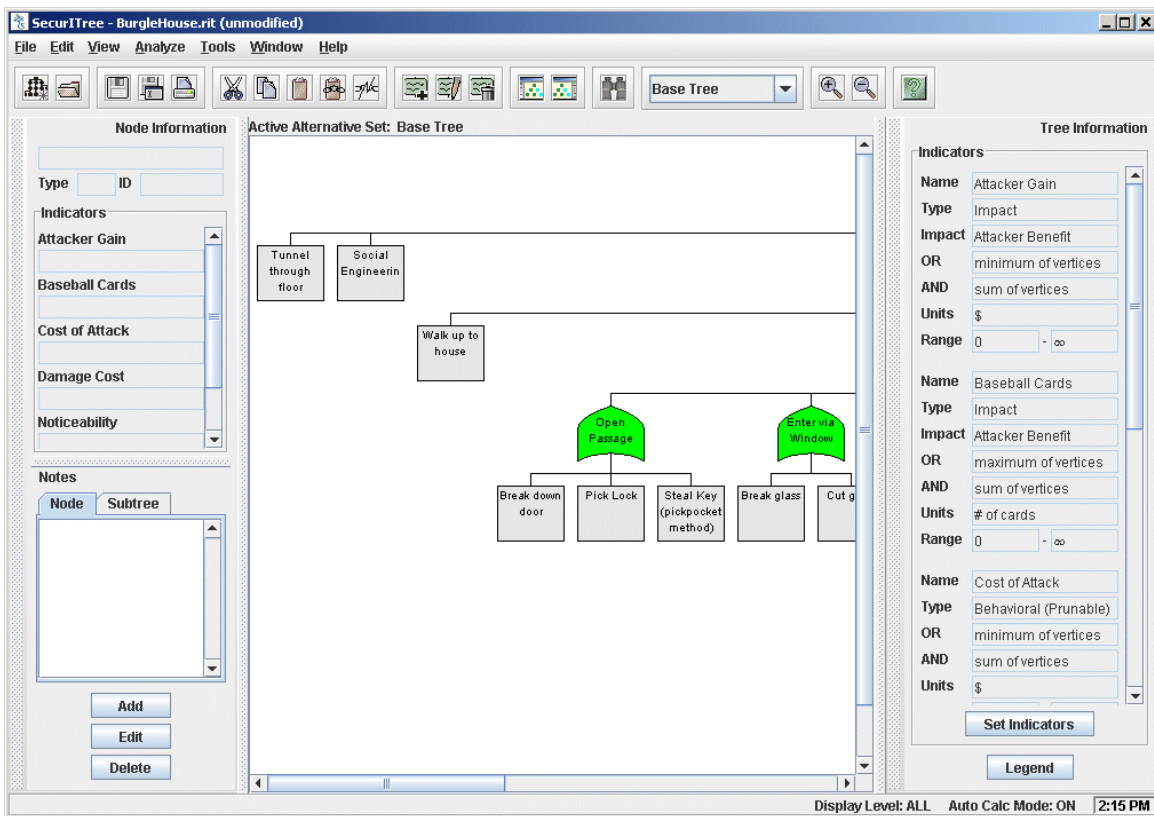


Open the BurgleHouse example file by clicking on *File* and selecting *Open Tree*. When the dialog appears, highlight *BurgleHouse.rit* and click *Open* (see **Figure 2**).



**Figure 2**

After a few seconds the sample attack tree will appear in the center panel of the SecurTree window (see **Figure 3**). This tree represents different ways in which a typical suburban home could be burglarized. Depending on the resolution of your computer screen, and the *zoom* level setting in SecurTree, it may not be possible to see the entire tree. The scroll bar at the bottom of the center panel can be used to see portions of the tree which do not fit within the window. Place your cursor on the blue scroll bar, press and hold the left mouse button and slide the mouse left or right until the desired portion of the image is visible in the panel. The tree can also be moved by placing the cursor on the center pane, depressing and holding the mouse button down (a will appear) while moving the tree to the desired position).



**Figure 3**





# Amenaza

TECHNOLOGIES LIMITED

After adjusting the scroll bar so as to make the topmost shape or node (i.e., the green shape entitled *Burgle House*) in the tree visible, place your cursor on it and single click your left mouse button. The *Burgle House* box will turn yellow to show that it is highlighted. Notice that the leftmost panel in the window, labeled *Node Information*, now contains information about the node. Since this node is the topmost or “root” node in the tree, it represents the ultimate goal of an attacker. In this case, the goal is to burgle the house.

The *Node Information* panel is handy for a quick glance at a node’s characteristics.

For a closer examination or to edit the node, you may find it easier to create a separate window. Do this by double clicking on the highlighted node. A window similar to that shown in **Figure 4** will appear.

This window contains the name or title of the node, *Burgle House*. In addition to the *Type* and *Indicator* information that will be discussed later, the section at the bottom of the window contains textual notes about the node.

For your convenience, different kinds of comments can be organized as different *Note* types. The *Note* types used in this example are *Node* and *Subtree*. Although we will not explore the option in this exercise, it is possible to define other *Note* types. For example, an analyst might want to record where the information used to create a node was found. This would be possible by defining a new *Note* type called *References*.

In **Figure 4** the *Note* type is set to *Node*. This means that the comments which were typed into the comment field by the analyst under this category should pertain to only this node.

Click on the *Subtree* tab to display *Subtree* notes. These comments apply to this node and all nodes below. Since this is the topmost or root node the *Subtree* notes apply to the entire tree. As

**Edit Node**

Name: Burgle House

Type: OR

**Behavioral Indicators**

|                   |       |           |        |
|-------------------|-------|-----------|--------|
| Cost of Attack    | \$    | [0 - ∞]   | 1      |
| Noticeability     | %/100 | [0 - 1]   | 0.0199 |
| Technical Ability |       | [1 - 100] | 3      |

**Impact Indicators**

|                                | Children's Impact | Impact Operator | Node's Impact | Impact Value |
|--------------------------------|-------------------|-----------------|---------------|--------------|
| Attacker Gain (AB) \$          | [0 - ∞] 0.0       | +               | 5,000         | 5,000        |
| Baseball Cards (AB) # of cards | [0 - ∞] 400       | +               | 100           | 500          |
| Damage Cost (VI) \$            | [0 - ∞] 0.0       | +               | 15,000        | 15,000       |

Change Node Color

**Notes**

Node \* Subtree \* Edit Notes

28 September 2005 - TR1  
There are various types of notes. Some comments are specifically for the node, others pertain to the subtree below.  
Hint: Change the "Note Type" to "Subtree" for further information about this tree.

OK Apply Cancel

**Figure 4**



in the previous case, it is entirely the analyst's responsibility to ensure that the right kind of comments are typed into each *Note* type.

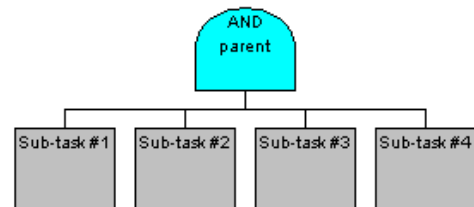
Please read the notes relating to the *BurgleHouse* node (in both *Node* and *Subtree* categories) to acquaint yourself with the tree model used in this exercise. Click *Cancel* to close the *Edit Node* window when you are ready to proceed.

Back at the main window (**Figure 3**) we redirect your attention to the rightmost panel. It is labeled, *Tree Information*. This panel, not surprisingly, provides information that applies to the entire tree. Click on the *Legend* button to display an information panel (**Figure 5**) showing the meaning of the various types of nodes. (Click *Close* to dismiss the *Legend* panel).

**Figure 5**

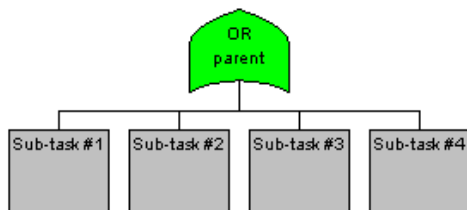
Nodes (boxes of various shapes) in an attack tree represent goals or states that an attacker wishes to achieve. A major premise in an attack tree model is that insight can be achieved by decomposing the high level parent goals into the smaller subtasks needed to achieve them.

Nodes below a particular node represent subtasks and are referred to as *children*. Conversely, the node above a given node is referred to as a *parent*. The nodes two levels above is called *grandparent* and so on. In **Figure 6** the grey, rectangular boxes (labeled as subtasks #1 - #4) are children of the cyan, round topped parent node.



**Figure 6**

**If all of the child subtasks beneath a parent must be achieved in order to realize the goal, then the parent is called an AND node.** Secur//Tree displays this as a cyan round topped box shape (**Figure 6**).



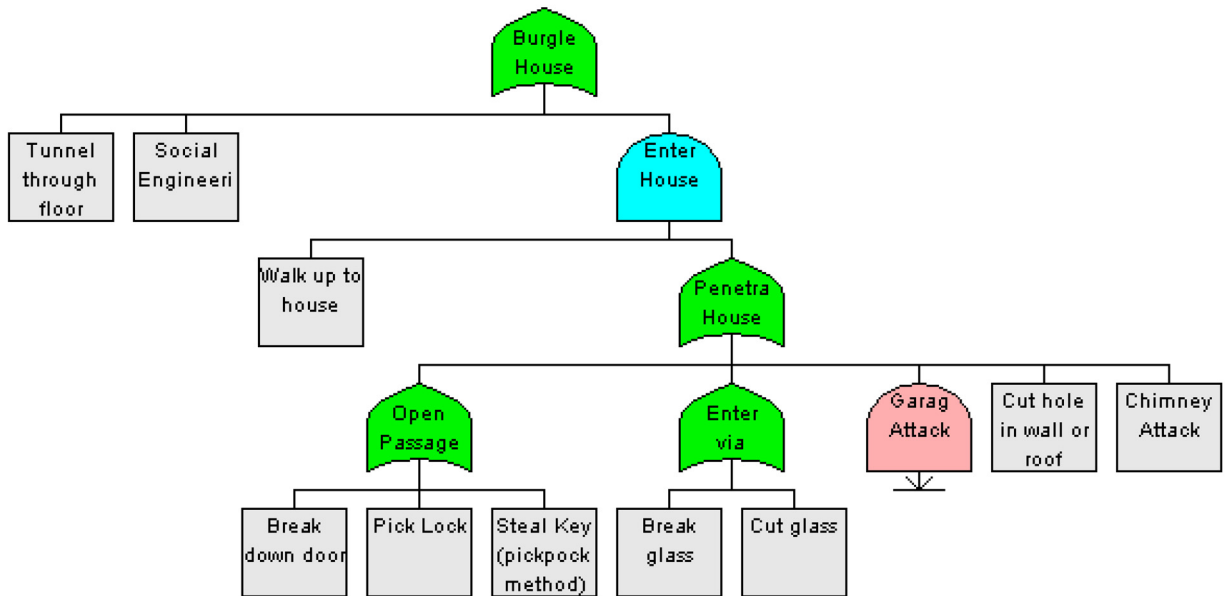
**Figure 7**

**In other cases, successfully performing any of the subtasks will cause the parent goal (known as an OR node) to be achieved.** Secur//Tree uses a green peaked shape with a concave bottom (**Figure 7**) to represent OR nodes.



The decomposition of tasks and goals into smaller components can continue to any desired level (each goal being represented by a separate node). At some point, however, the analyst decides that the level of detail in a



node is sufficient. These nodes are considered to be *atomic*<sup>3</sup>. That is, the description of the task is precise enough for someone skilled in the art to perform the activity. These atomic nodes are known as *leaf* nodes and are represented by Secur/Tree as square cornered rectangles (shown in both **Figure 6** and **Figure 7**). *Leaf* nodes represent activities that can be performed by adversaries. *AND* and *OR* nodes represent the states achieved as a result of these activities.



**Figure 8**

The information displayed in the Secur/Tree main window (**Figure 3**) should now begin to make sense. Use the  and  magnifying glasses found on the toolbar at the top of the window to zoom in and out as required.

Three approaches are shown to *Burgle the House* (**Figure 8**). Burglars may go underground and *Tunnel through the floor*, they may use *Social Engineering* to pretend to be someone trusted (perhaps a repairman) or they may attempt to *Enter House* by approaching the house on foot and then overcoming the house's defenses<sup>4</sup>.

<sup>3</sup> We told you this was a powerful tool! However, our use of the term *atomic* refers to the original Greek meaning of the word *atomos* (indivisible). In this context *atomic* simply means that the goal cannot be subdivided.

<sup>4</sup> The *Walk up to house* node (and its *AND* parent, *Enter House*) could easily have been omitted, leaving the *Penetrate House* subtree positioned directly below the *Burgle House* root node. Adding the apparently redundant intermediate node makes it easier to enhance the model later to include perimeter defenses such as fences.



# Amenaza

TECHNOLOGIES LIMITED

The five nodes directly below the *Penetrate House* node represent possible burglar tactics. Double click on some of these nodes and read the descriptions of the attacks. *Cancel* out of the node edit windows when your curiosity is satisfied.

| Indicators |                            |
|------------|----------------------------|
| Name       | Cost of Attack             |
| Type       | Behavioral (Prunable)      |
| OR         | minimum of vertices        |
| AND        | sum of vertices            |
| Units      | \$                         |
| Range      | 0 - ∞                      |
| Name       | Damage Cost                |
| Type       | Impact                     |
| Impact     | Victim Impact              |
| OR         | minimum of vertices        |
| AND        | sum of vertices            |
| Units      | \$                         |
| Range      | 0 - ∞                      |
| Name       | Noticeability              |
| Type       | Behavioral (Prunable)      |
| OR         | minimum of vertices        |
| AND        | $1 - [(1-a)(1-b)...(1-n)]$ |
| Units      | %/100                      |
| Range      | 0 - 1                      |

**Set Indicators**

**Figure 9**

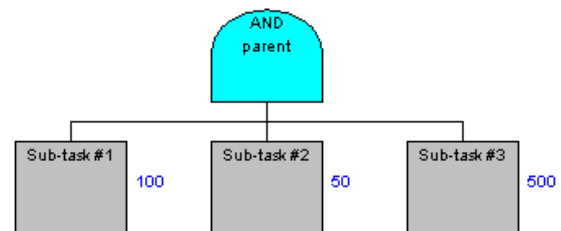
**10** shows an *AND* node with three subtasks. The costs of carrying out the subtasks have been defined as \$100, \$50 and \$500. The cost of arriving at the parent (*AND*) node given these particular subtask costs is \$650 since all three subtasks must occur and the cost will be the sum of costs of the subtasks. This is correctly shown by the *Sum of Vertices* formula associated with *Cost of Attack AND* nodes (**Figure 9**).

Returning your attention to the main window, we continue our examination of the *Tree Information* panel. You will notice that a number of *Indicator Functions* have been defined (**Figure 9**). Indicators functions are of two types: *Behavioral* (also known as *Prunable*) and *Impact*. If you scroll the panel shown in **Figure 9** you will see behavioral indicators for *Noticeability*, *Cost of Attack* and *Technical Ability*. *Damage Cost* and *Attacker Benefit* and *Baseball Cards* are of type *Impact*. *Impact Indicators* will be discussed later in the tutorial.

*Behavioral Indicators* are factors which influence a person's behavior. They have some bearing on whether or not an attacker is willing or capable of carrying out an attack. **SecurITree allows you to define indicators of your own choosing, but we suggest you stick with these three until you gain experience.**

Each *Behavioral Indicator* has associated with it two mathematical formulae – one for *AND* nodes and the other for *OR* nodes. They reflect how the resources required to carry out subtasks relate to the resources required to reach a parent state. The particular mathematical formula chosen for use with *AND* and *OR* nodes depends on the nature of the indicator. SecurITree allows the analyst to select from a predefined list of formulae that have been found useful for attack tree analysis.

For example, **Figure**



**Figure 10**

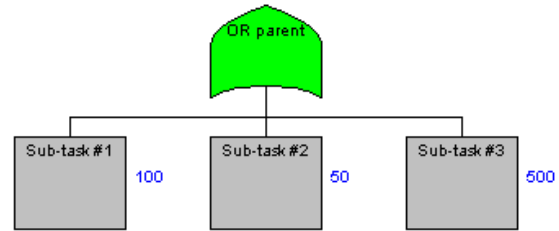




# Amenaza

TECHNOLOGIES LIMITED

If the parent node were changed to an *OR* node, as shown in **Figure 11**, the minimum or least costly way of achieving the parent node is through subtask #2 (\$50). The formula associated with *Cost of Attack OR* nodes is seen to be (**Figure 9**) *minimum of vertices*.



**Figure 11**

Different behavioral indicators have different mathematical functions associated with them. They all cause the resource requirements

defined at the lowermost leaf nodes of the tree (the atomic attacks) to be passed upward to the ultimate attack goal. Formulae are chosen such that the indicator values at any point along the tree represent the resources required to reach that point in the tree along a given attack path.

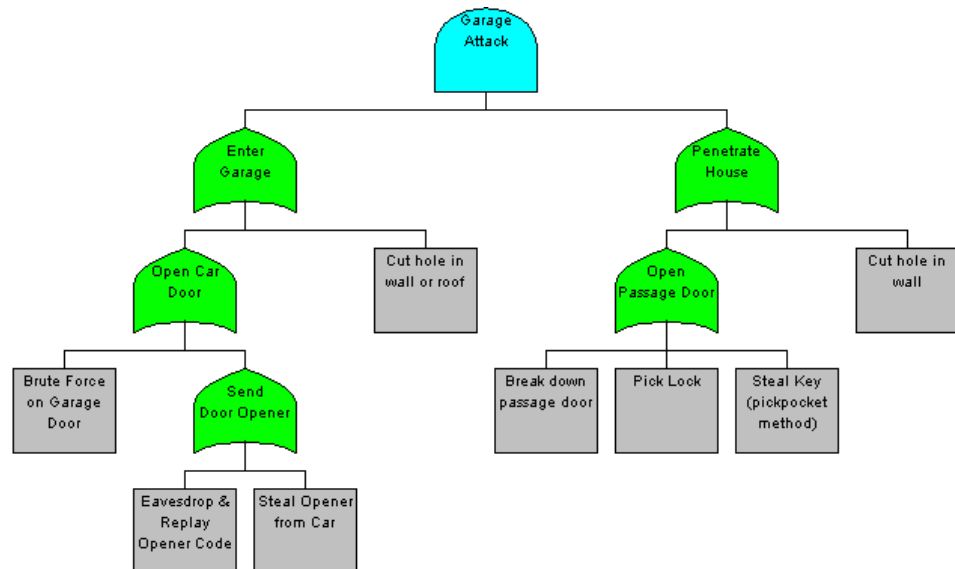
Note that the non-leaf values are calculated independently for each behavioral indicator. This means that the set of behavioral indicator values calculated at any particular non-leaf node may reflect several different paths through the tree. This will be made explained further when we consider *attack scenarios*.

Locate the salmon colored node labeled *Garage Attack*. This node is *rolled up*. That is, there is detail below this point which has been hidden to conserve space on the screen. Right click the *Garage Attack* node and select *Roll Down Subtree* from the popup menu. The node should expand to show more detail (see **Figure 12**).

Double click on the leaf node *Eavesdrop & Replay Opener Code* (in the bottom left portion of the tree that deals with opening the car door) to bring up a node *Edit* window.

Examine the values in the *Behavioral Indicators* section of the *Edit* window (see **Figure 13**). An explanation of why the values were chosen is found in the *Node* notes

section of the window (scroll down as required). In essence, the behavioral indicator values assigned to the various leaf nodes are educated estimates made by an expert in the field. It is



**Figure 12**



probably impossible to find definitive values for the indicators, but the estimates can be at least the correct order of magnitude. For example, an attack that nominally costs \$5,000 could conceivably be done for \$4,000 or might cost as much as \$6,000. We would be fairly certain, however, that it would not cost \$50,000 (or \$500). **The leaf node indicator values are one of two major assumptions upon which attack tree analysis is based.**

**Edit Node**

Name:

Type:

**Behavioral Indicators**

Cost of Attack \$ [0 - ∞]

Noticeability %/100 [0 - 1]

Technical Ability [1 - 100]

**Impact Indicators**

Attacker Gain (AB) \$ [0 - ∞]

Baseball Cards (AB) # of cards [0 - ∞]

Damage Cost (VI) \$ [0 - ∞]

**Notes**

Node \* Subtree \* Edit Notes

Each time the homeowner uses their garage door opener, a set of digital codes is transmitted via short-range radio. There are several problems to picking this up. Recent door openers operate at very low power levels that make it difficult to eavesdrop from further than a block away. The door opener system also uses a

**Figure 13**

**Once the *threat agents* have been identified, it is essential to list their traits.** Specifically, we need to know what resources they have at their disposal compared to the indicators used in the model. In our example, the behavioral indicators are *Noticeability*, *Cost of Attack* and *Technical Ability*.

Cat burglars represent a skilled and sophisticated variety of house robber. They work quietly, usually at night, and are skilled in the art of opening locks, breaking windows silently, defeating burglar alarms and at bypassing all sorts of security systems without being detected. They are career criminals who really do not want to get caught – judges take a dim view of their endeavors. They probably view crime as a business – and nothing ruins profitability like a stint in the slammer! It is worthwhile for them to make an investment in tools if there will be returns down the road.

Cancel out of the *Edit* window. Single click your cursor on the parent, grandparent and great-grandparent nodes above *Eavesdrop Code*. As each node is highlighted, watch the indicator values in the *Node Information* window.

It is possible to see which nodes are easy or difficult to attain by simply clicking your way around the tree. However, this is very tedious and prone to error. It also does not lend very much insight as to whether or not the attack is practical and if so, by whom.

To answer these questions, we introduce the concept of *threat agents*. **Threat agents are classes of people who have a motivation to attack a system.** For this example we will only consider two kinds of threat agents (but you can certainly think up many others):

- cat burglars
- juvenile delinquents

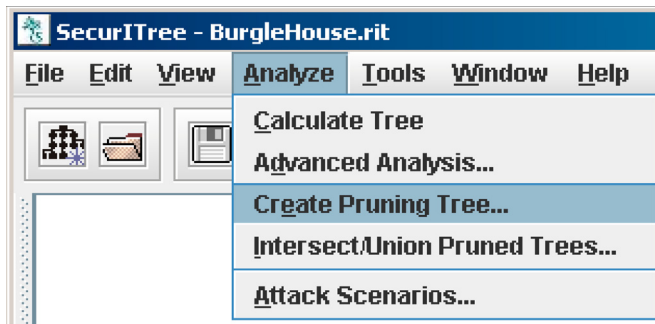
**Once the *threat agents* have been**



Juvenile delinquents, on the other hand, tend to have all of the subtlety of a bull in a china shop. They like to break things and do not worry much about getting caught. Whether this is because of their troubled childhood or lax young offender legislation is a matter for another forum to debate. They normally do not spend much of their own money during the commission of a crime. Based on these traits, we can estimate approximate numeric values. These are shown in the following *Threat Agent Resource* chart:

| <i>Indicator</i><br><i>Threat Agent</i> | <b>Cost of Attack</b> | <b>Tolerance for being Noticed</b> | <b>Technical Ability</b> |
|---|-----------------------|------------------------------------|--------------------------|
| Cat Burglar                             | \$10000               | 10%                                | 70 (out of 100)          |
| Juvenile Delinquent                     | \$50                  | 30%                                | 20 (out of 100)          |

The values chosen are essentially the opinions of an expert in the field. They are intended to convey the **magnitude** of the resources available to the threat agent. The reasonableness of these assumptions is key to getting useful predictions from Secur/Tree. We will now show how Secur/Tree allows you to determine which house burglary attacks are possible for cat burglars and juvenile delinquents.

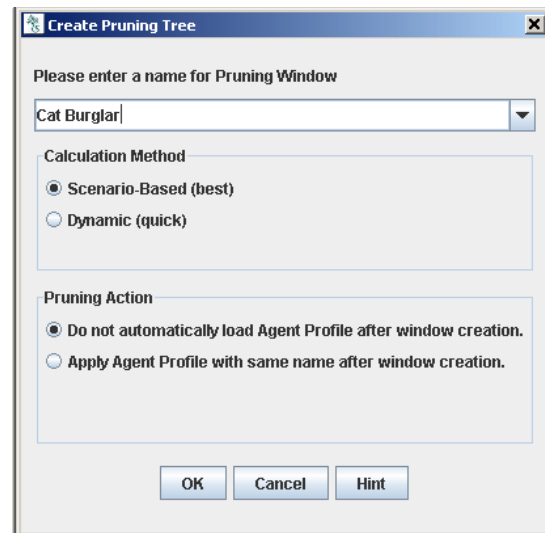


**Figure 14**

For each type of threat agent being considered, a working window with a copy of the attack tree model we have defined must be created. This working window

allows us to selectively remove branches of the tree that cannot be reached by a particular threat agent. Hence, it is called a *pruning window*.

To create a pruning window, click on *Analyze* > *Create Pruning Tree* (as shown in **Figure 14**). After a few seconds a dialog box will appear prompting you for the name of the *pruning window*. You can choose any name you like, but it is usually convenient to use the name of the *threat agent* whose behavior is being analyzed. In our case, we are interested in a *Cat Burglar*, so enter the name, *Cat Burglar*, in the name field (see **Figure 15**) and click *OK*. A new window will appear. This window contains a working copy of the house burglary attack tree. Depending on the size

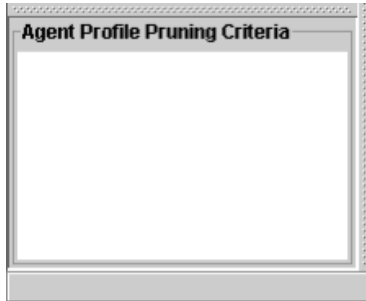


**Figure 15**




and resolution of your computer monitor you may need to resize the window somewhat and/or zoom in or out to get a comfortable view of the attack tree.

In order to see which attacks are possible for a particular threat agent (in this case a *Cat Burglar*) it is necessary to describe their capabilities.



**Figure 16**

The *Agent Profile Pruning Criteria* panel in the bottom of the new window (shown in **Figure 16**) provides a display of the resource constraints that are placed upon the *Cat Burglar* threat agent. It is initially empty since no filtering criteria have been defined yet.

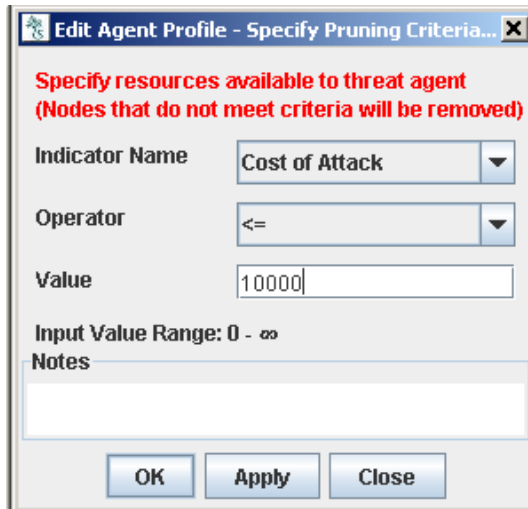
From the toolbar at the top of the pruning window (**Figure 17**), click on the  to open a dialog window (shown in **Figure 18**) for defining the resources available to the threat agent.

Attacks requiring more resources than are available to the *Cat Burglar* will be removed or *pruned* from the working copy of the attack tree.

Create an *Agent Profile* for the *Cat Burglar* by entering the values in the *threat agent* resource chart we created earlier via the window shown on page 11.



**Figure 17**



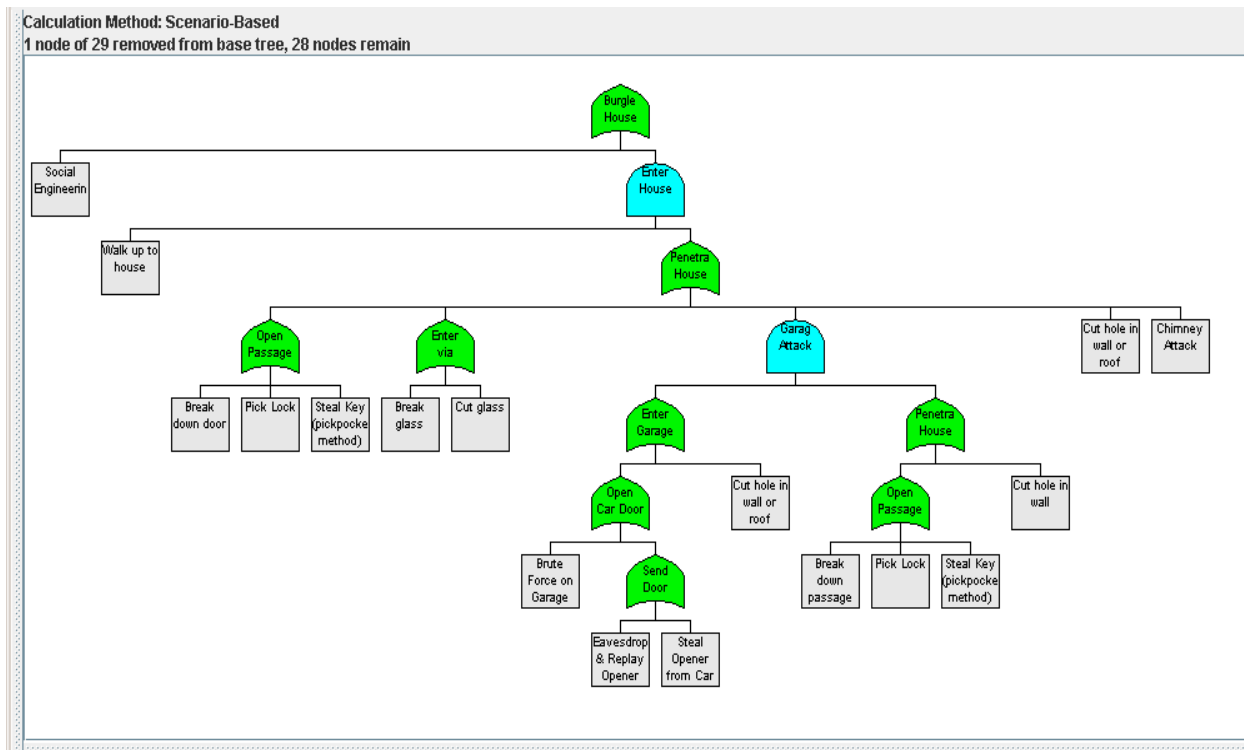
**Figure 18**

The first constraint to be added to the profile is the amount of money the *Cat Burglar* is willing to spend (for tools, research, etc.) on a house burglary. From our table we see that we believe the upper limit is about \$10,000. Ensure that the *Indicator Name* (in **Figure 18**) is set to *Cost of Attack*. (If necessary, click on the ▼ in the *Indicator Name* pulldown and select *Cost of Attack*.) Then, choose the <= (less than or equal to) operator by clicking on the ▼ in the *Operator* pulldown. Finally, enter the value of 10000 in the *Value* field (type 10000 without the “\$” or the “,”). When all of the parameters are set correctly, click *Apply*. This operation has told Secur//Tree that all of the attacks requiring more than \$10,000 to carry out should be removed from the working copy of our model. If you were attentive, you may have noticed

that the tree depicted in the *Cat Burglar* window shimmied slightly when the *OK* button was pressed. In this case, only one node was removed – the attack involving the construction of an expensive tunnel. The remaining nodes are all attacks that are within the financial capability of



our intrepid cat burglar (see **Figure 19** – you may have to move the *Edit Agent Profile - Specify Pruning Criteria* window to one side in order to see the pruned tree).

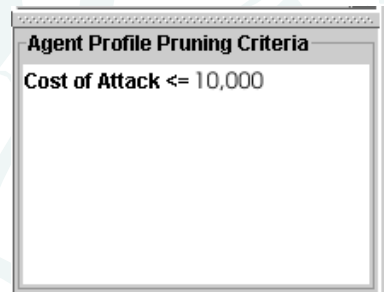


**Figure 19**

You may also have noticed that the *Agent Profile Pruning Criteria* panel in the lower section of the pruning window (**Figure 20**) now shows that the *Cat Burglar* is only able to spend up to and including \$10,000.

The choice of \$10,000 is an assumption. It certainly is not exact and may even be wrong. It is, however, a defensible estimate based on informed opinion. We believe that \$10,000 is of the right order of magnitude. We can be pretty certain that our cat burglar will not spend \$50,000 on the attack and we can be very certain that they are not able or willing to spend \$500,000.

In a real analysis it would be worthwhile to adjust this value up or down to see if the results change. Significant changes in the results may indicate that the model is sensitive to changes in the parameter being varied. In these situations both the leaf node indicator values and the capabilities of the threat agents should be scrutinized to ensure that they represent accurate



**Figure 20**



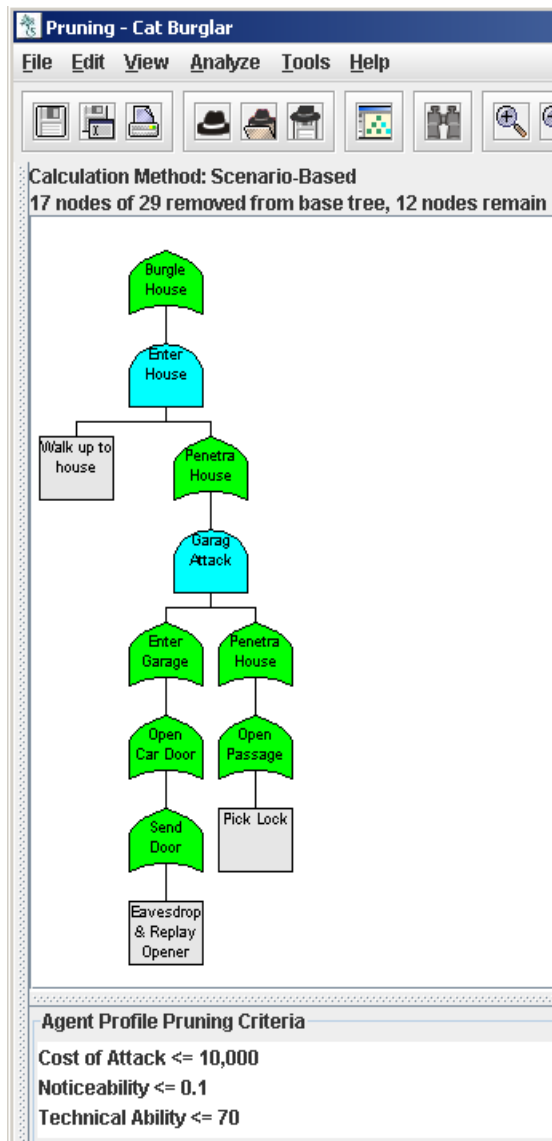


# Amenaza

TECHNOLOGIES LIMITED

estimates. SecurITree is capable of producing special *sensitivity analysis* reports to help you assess the strength of your assumptions. This sort of “**what if**” type of analysis is easy with SecurITree.

Now try pruning the *Cat Burglar* tree on the remaining two indicator criteria, *Noticeability* and *Technical Ability*. Using the same technique as with *Cost of Attack*, choose *Noticeability* from the drop down list as the *Indicator Name*. The *Operator* should be set to  $\leq$  and the value set to *0.1* (suggesting that the Cat Burglar will avoid attacks that involve a risk of getting caught greater than 10%). Click *Apply* to prune on the *Catchability* criterion.



**Figure 21** – Attacks Available to Cat Burglar

In the same fashion, set the *Technical Ability* criteria to be  $\leq 70$  (out of 100). This shows that our Cat Burglar is highly skilled. Once all three pruning criteria have been specified, dismiss the *Edit Agent Profile - Specify Pruning Criteria - Cat Burglar* window by pressing *Close*.

After pruning on all three criteria, your pruned tree should look like **Figure 21**. More than half the tree (17 out of 29 nodes) has been removed as being unavailable to the Cat Burglar. **In other words, the remaining nodes are the points that need protection against Cat Burglars.**



Interestingly, only three of the remaining nodes are *leaves*. Attacks always commence at the leaf nodes. **If we are able to prevent the attacks represented by these three leaf nodes then a Cat Burglar will be unable to break into the house.**

Why are so many attacks beyond the ability of a supposedly highly skilled Cat Burglar? The Cat Burglar’s weakness is his or her unwillingness to be caught. Although we cannot think of any easy way to detect the eavesdropping of radio waves, it is possible that the addition of a good burglar alarm inside the garage would improve the odds of detecting an attack on the passage door. This, in turn, might be enough to dissuade the Cat Burglar from trying a garage attack.

Before closing the Cat Burglar window, you should save the criteria you keyed in. If the *threat agent* traits listed in the *Agent Profile* are saved in a file,



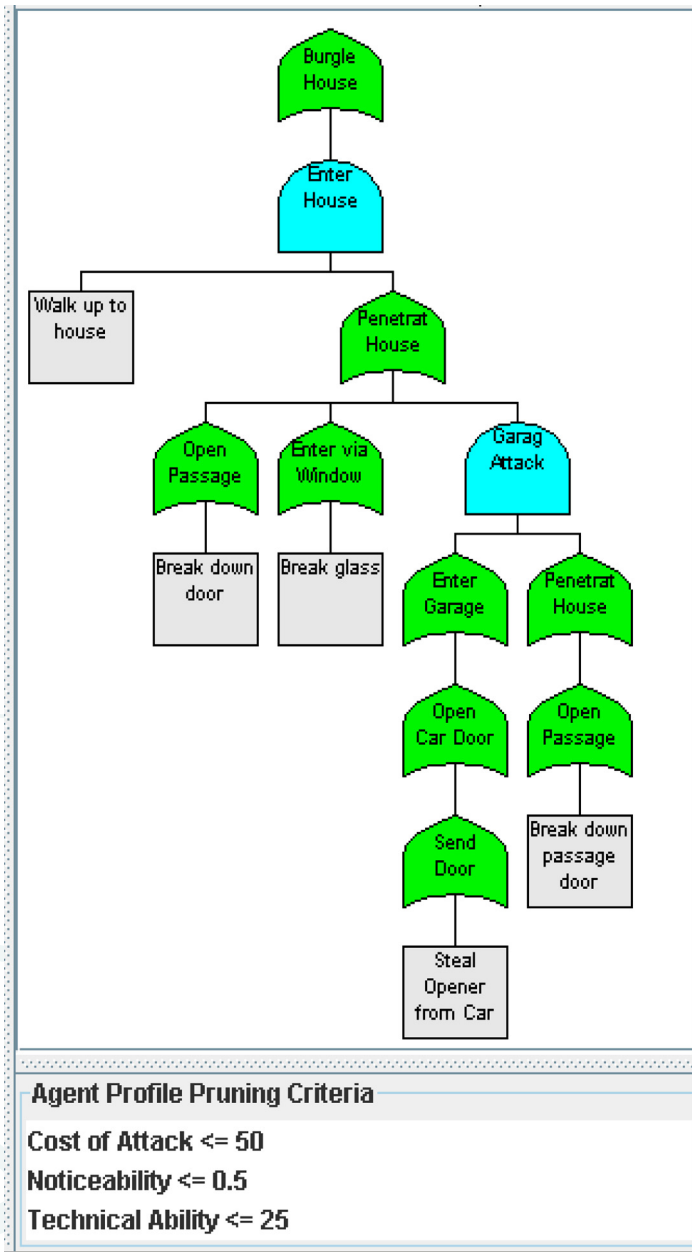
less typing is required to repeat the pruning operation after each improvement to the system is modeled. **The ability to save *threat agent* characteristics makes it easy to explore the effect of design changes to the system being modeled.**

To save the *Cat Burglar Agent Profile* to a file, click on the  button located on the *Cat Burglar* pruning window toolbar. When the dialog window appears, save the profile using the default name (probably *Cat Burglar.agt*) in the folder or directory of your choosing. You may now close the *Cat Burglar* pruning window by clicking on the  in the upper right corner.

Bring the main Secur/Tree window (labeled *SecurTree - BurgleHouse.rit*) to the foreground by clicking on it. (You may have to move or resize the *Cat Burglar* pruning window in order to expose the main window.) Now, using the same technique as before, create another pruning window called *Juvenile Delinquent*. Following the same general procedure as with the *Cat Burglar* (described on page 12), set the *Noticeability*  $\leq 0.3$ , *Cost of Attack*  $\leq \$50$ , *Technical Ability*  $\leq 20$  as shown in the table earlier.

The resulting tree (**Figure 22**) shows the attacks possible by the *Juvenile Delinquent*. Note the preponderance of the word “break” in these attacks. Once you realize that stealing the garage door opener probably involves breaking a car window and grabbing the remote, then the pattern of attack is unmistakable.

Protecting against this sort of adversary will generally involve making the physical access points stronger. Burglar bars over the windows. Steel doors with steel reinforced door jams are the techniques of choice. Not leaving the garage door opener in the car would help too.



**Figure 22** – Attacks Avail. to Juvenile Del.



Close the Juvenile Delinquent window and return to the main window. We are now going to edit the original tree. Before editing, it is necessary to ensure that all of the pruning windows have been completely closed (and not just minimized or buried beneath another window). On the main window, click on *Window > Close All Analysis Windows*.

One approach to make it harder to break into the house is to identify all of the *leaf node* vulnerabilities that are easily exploited and apply suitable countermeasures (such as the burglar bars mentioned earlier). This tends to be a lot of work because there may be many leaf nodes that require attention. Also, if we have forgotten to include one or more vulnerabilities then they will not be addressed.

A better idea is to make an architectural change that will protect multiple vulnerabilities. As long as our countermeasure protects against all *classes* of vulnerabilities we should be safe since

exploiting any of the vulnerabilities within a class usually requires the adversary to possess similar skills and resources.

We would like to know whether or not adding a perimeter fence around the house will be an effective protection. The kind of fence we are thinking about is 8 ft (2.4 m) high, reinforced chain link, with coils of razor wire and an electronic alarm system to detect attempts to cut through it.

Locate the *Walk up to house* node that is located below the *Enter House AND* node. Double click on the *Walk up to*

Figure 23



house node to enter the edit dialog. Change the name of the node to be *Penetrate Perimeter Fence* and set the node type to be *OR* (as shown in **Figure 23**). Also update the *Subtree* notes if you like. Press *OK* to continue. Acknowledge the warning that *Indicator values for node will be removed . . .* by clicking *OK*. Ignore and dismiss any error messages you may get advising that Secur/Free is unable to calculate the tree because the *Penetrate Perimeter* node has no values defined. These messages will cease once the new subtree structure is complete.

Ensure that the *Penetrate Perimeter* node is still highlighted and click on the  icon on the

**Add Node**

Name: High Tech Attack

Type: LEAF

**Behavioral Indicators**

|                   |       |           |       |
|-------------------|-------|-----------|-------|
| Cost of Attack    | \$    | [0 - ∞]   | 2,000 |
| Noticeability     | %/100 | [0 - 1]   | 0.1   |
| Technical Ability |       | [1 - 100] | 80    |

**Impact Indicators**

|                     |            |         |     |
|---------------------|------------|---------|-----|
| Attacker Gain (AB)  | \$         | [0 - ∞] | 0.0 |
| Baseball Cards (AB) | # of cards | [0 - ∞] | 0.0 |
| Damage Cost (VI)    | \$         | [0 - ∞] | 200 |

Change Node Color

**Notes**

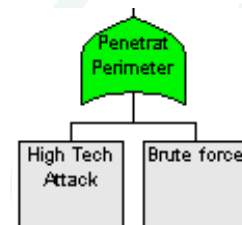
Node \* | Subtree | Edit Notes

Suppress alarm systems then cut fence.

OK | Apply | Cancel

toolbar to add a child below *Penetrate Perimeter*. When the *Add Node* dialog box appears, fill in the values as shown in **Figure 24** and press *OK*.

Repeat this operation to add a second leaf node child to *Penetrate Perimeter*. The name of the second child should be *Brute Force*. Set the indicator values to be: *Cost of Attack – 50, Noticeability – 0.95, Technical Ability – 15, Attacker Gain – 0, Baseball Cards – 0, Damage Cost – 500*. After pressing *OK* to save the change the resulting subtree should look like **Figure 25**




**Figure 25**

We will now see if the fence countermeasure has affected the Cat Burglar's ability to attack the house.


As before, create a pruning window and label it *Cat Burglar*. Use the same *Analyze > Create Pruning Tree* operation

**Figure 24** – Add *High Tech Attack* dialog



shown earlier in **Figure 14** and **Figure 15**. This time instead of reentering the Cat Burglar characteristics, reload them from the profile you created earlier. This is done by clicking on the  icon on the pruning window toolbar. When the file selection dialog appears, make certain you select the Cat Burglar agent profile you created earlier (probably `Cat_Burglar.agt`). Complete the operation by clicking *Open*.

This time you should have been given a warning message that the *Root node was removed ....* After you dismiss the warning (press *OK*) you are left with a blank pruning window. This means that there are no paths leading from the leaf nodes to the root of the tree within the capability of the Cat Burglar. The countermeasure is effective.

If you wish to save your revised tree we recommend that you do so under a new file name as we will be using the original tree for the next example. To save the revised tree, ensure that all of your pruning windows are closed and click on the  icon on the toolbar. Provide an appropriate file name when so prompted.

## Impact Analysis

Risk is a combination of the likelihood that an incident will occur and the damage that it will cause. Secur//Tree's capabilities-based attack tree analysis combines threat and vulnerabilities to predict the behavior of different types of attackers, thus providing a measure of a particular attack's likelihood. However, this only represents half of the risk equation. In order to be a complete risk analysis solution, Secur//Tree must also be able to model the projected impacts on the victim (as well as potential benefits to the attacker).

The impact of an attack is very system and system owner specific. For example, the impact of a house burglary will depend heavily on what is in the house. Two identical houses might have quite different contents. This loss is independent of how the burglar enters the home.

In addition to the impact occasioned by the attacker accomplishing their overall or root goal, there are also impacts associated with the particular attack used. In the previous analysis, we saw that the home could be entered by brute force attacks which would damage doors and windows or even walls. In addition to replacing the stolen items a victim would also bear the cost of the repairs. So, given a choice, it is better to be robbed by a skilled lockpick (who causes no damage gaining entry) than by a brute force thug. Secur//Tree represents these concepts via *Impact Indicators*. *Impact Indicators* work very much like the *Behavioral Indicators* we examined earlier. However, there is one important difference. The *behavioral indicator* values of non-leaf nodes are calculated using only the values defined in the leaf nodes. *Impact indicators* use a similar calculation mechanism, but also allow the analyst to directly influence values of non-leaf nodes.

Secur//Tree can also use *impact indicators* to represent the gains that an attacker will get from carrying out an attack. Different types of gains motivate different attackers to a greater or lesser degree.





# Amenaza

TECHNOLOGIES LIMITED

Ensure that you are working with the original `BurgleHouse.rtf` tree by closing the tree (use `File > Close` as found on the main window toolbar. If you are asked to save your work, ensure you do not overwrite the original file (answer *No* to the prompt). Reopen the `BurgleHouse.rtf` exactly as instructed at the beginning of this tutorial.

Examine the *Pick Lock* node (located under the *Open Front/Back Door* subtree. Note that the *Damage Cost* for picking a lock has been set to \$0. Then, take a look at the node's sibling, *Break Down Door*. In this case, the *Damage Cost* has been set to \$250 reflecting how much it will cost a homeowner to replace the door that has been physically damaged. These values are passed upwards through the tree based on the defined *Damage Cost* indicator functions. Now examine the tree's root node, *Burgle House* (shown in **Figure 26**). Notice that the section entitled *Impact Indicators* has more fields than it did when we examined leaf nodes. The additional fields allow the analyst to show that, no matter which method is used to gain access to the house, \$15,000 of valuables will be stolen or destroyed. This cost is in addition to whatever damage is caused by the attacker as they enter the building.

**Figure 27** shows that the *Children's Impact* is 0 – i.e., the computed value passed up from below is \$0. The *Impact Operator* is set to +. This means that the *Children's Impact* value will be added to the *Node's Impact* value which has been directly entered by the analyst. The result is the total *Impact Value* (which is  $\$0 + \$15,000 = \$15,000$ ). It is often the case that the largest business impacts occur at the mid or top levels of an attack tree.

Analysts can enter information derived from interviews or other sources to show the effect the attack has at any level in the tree. In the house burglary tree, another example is shown in the *Garage Attack AND* node. In this case, the analyst speculates that if the thief enters through the garage he will notice the tools, bicycles and skis (which have a total value of \$3,000) and steal them. These items (in our model) would have been overlooked by the thief if other methods of entry were used. Therefore, garage-based house attacks are more expensive for the victim.

The screenshot shows the 'Edit Node' dialog box for the 'Burgle House' node. The dialog is divided into several sections:

- Name:** Burgle House
- Type:** OR
- Behavioral Indicators:**
  - Cost of Attack: [0 - ∞] 1
  - Noticeability: [0 - 1] 0.01
  - Technical Ability: [1 - 100] 2
- Impact Indicators:**

|                       | Children's Impact | Impact Operator | Node's Impact | Impact Value |
|-----------------------|-------------------|-----------------|---------------|--------------|
| Attacker Gain [0 - ∞] | 0.0               | +               | 5,000         | 5,000        |
| Damage Cost [0 - ∞]   | 0.0               | +               | 15,000        | 15,000       |
- Notes:**
  - Node \* Subtree \* Edit Notes
  - 28 September 2005 - TRI
  - There are various types of notes. Some comments are specifically for the node, others pertain to the subtree below.
  - Hint: Change the "Note Type" to "Subtree" for further information about this tree.

Buttons at the bottom: OK, Apply, Cancel.

**Figure 26**



# Amenaza

TECHNOLOGIES LIMITED

In addition to the *Damage Cost* impact indicator, you may have noticed that there are two other impact indicators, one called *Attacker Gain* and the other *Baseball Cards*. *Attacker gain* represents the monetary benefit that an attacker gets from carrying out an attack. In this simple example, we have supposed that the attacker will sell the stolen goods on the street for about one third the replacement value. *Baseball Cards* reflects a collection of baseball memorabilia the homeowner possesses. These cards are of little monetary value but are potentially interesting to a baseball fan – especially so to an adolescent. In this quick introduction to attack tree analysis we will not go into great detail about modeling attacker benefits. Secur/Tree is capable of very sophisticated analysis but that is beyond the scope of this introductory tutorial. Suffice it to say that attackers, like everyone else, make their decisions based on cost benefit analysis. They consider carefully the various resources they will need to expend for an expected reward. Only when the equation is in their favor will they act.

| Impact Indicators   |            | Children's Impact | Impact Operator | Node's Impact | Impact Value |
|---------------------|------------|-------------------|-----------------|---------------|--------------|
| Attacker Gain (AB)  | \$         | [0 - ∞] 0.0       | + ▼             | 5,000         | 5,000        |
| Baseball Cards (AB) | # of cards | [0 - ∞] 400       | + ▼             | 100           | 500          |
| Damage Cost (VI)    | \$         | [0 - ∞] 0.0       | + ▼             | 15,000        | 15,000       |

Figure 27

## Attack Scenarios

The previous examples have been somewhat vague in explaining how Secur/Tree determines what portions of the attack tree model are beyond the capabilities of particular adversaries. Initially you might think that Secur/Tree simply examines each node's set of indicator values and removes nodes that have resource values higher than those of the adversary. Conceptually this is correct, but it will not quite work.

The indicator values calculated at each node show the least cost path for each indicator from the tree's leaf nodes to the chosen node. Because each indicator is examined in isolation the indicator value almost always ends up being very low. Most indicators have a trivial route to a particular node if no other constraints are considered. The values shown in the *Node Information* panel are derived from a collection of inexpensive paths to that node. No single path necessarily corresponds to the set of values at a given node. The set of node values are of little benefit to the analyst.

The solution is to examine every possible path through the tree and calculate node values based on each specific path.

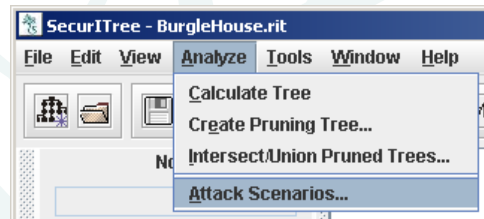


Figure 28 – Attack Scenario Pulldown Menu

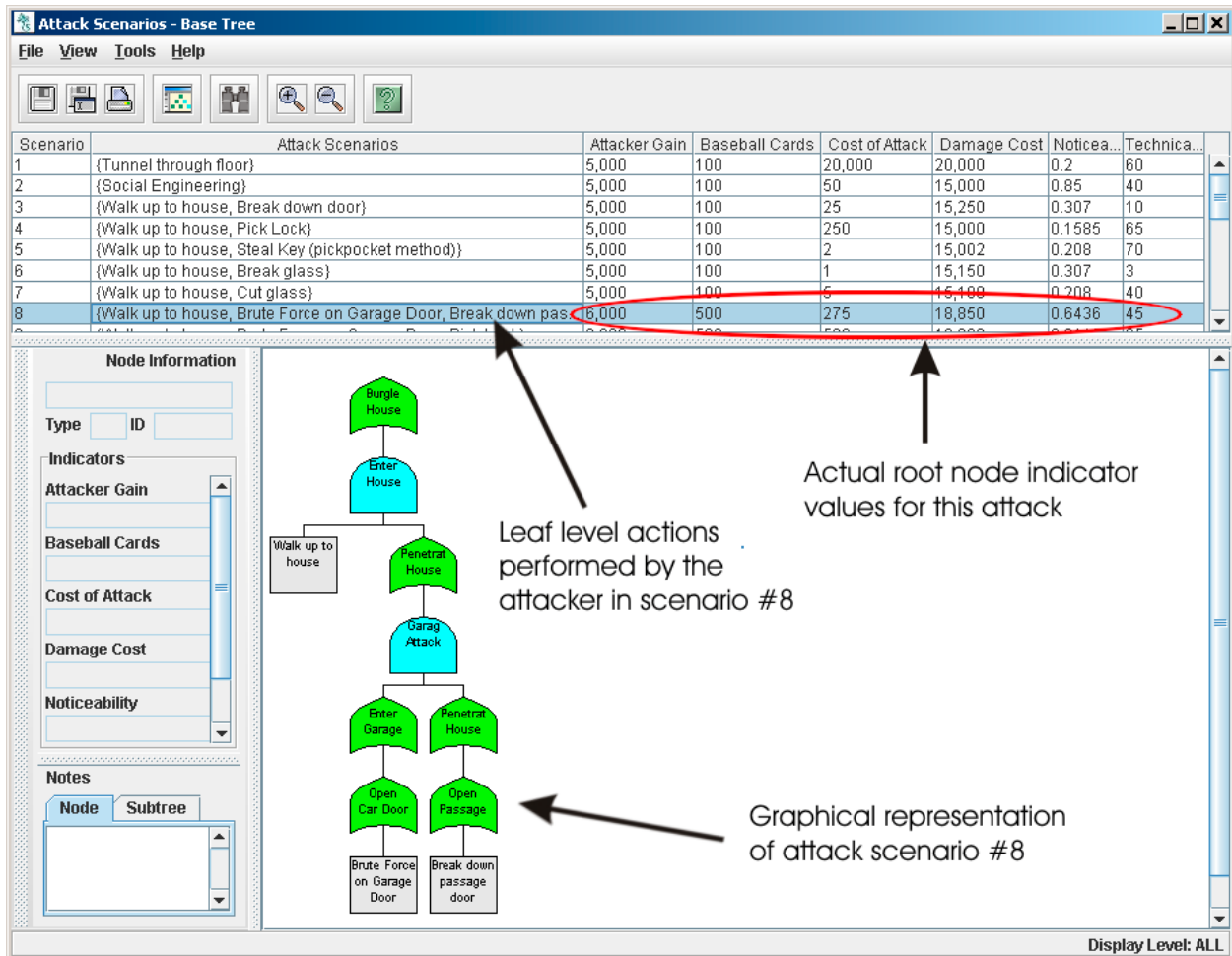


# Amenaza

TECHNOLOGIES LIMITED

Paths that have resource requirements beyond the reach of a particular adversary can be removed. This would be very tedious by hand, but is trivial using SecurTree.

After ensuring that all node editing windows are closed, click on `Analyze > Attack Scenarios` as shown in **Figure 28**. In a few moments, a new window will appear. The *attack scenario* window contains a list of each specific attack that can be used to reach the root node. The window (**Figure 29**) is divided into several sections. The topmost, horizontal panel lists each attack scenario. By clicking on a particular scenario (scenario #8 is shown in the example) the lower panel displays the path or combination of paths comprising that particular scenario.



**Figure 29 – Attack Scenarios for Entire House Burglary Tree**

The indicator values for the root node for that specific attack are also shown.

*Attack scenarios* are extremely important. In the case of *behavioral indicators* they show the resources required by an attacker for the particular scenario. For *impact indicators*, they allow the analyst to see the impact to the victim (and the benefits to the attacker) for each scenario.



# Amenaza

TECHNOLOGIES LIMITED

The leaf node events corresponding to each attack scenario could also be used to construct an attack detection system. By gathering and comparing information from intrusion sensors with the attack scenarios it is possible to determine when an attack is underway.

## Risk Estimation

Attack scenarios can be used to combine capabilities-based analysis and impact modeling to easily produce a prioritized list of risks in a system. To demonstrate this feature, ensure that all *attack scenario* windows are closed and that the only window open is the main SecurITree window.

Create a *Cat Burglar* pruning window (Analyze > Create Pruning Window) using the same procedure as was described earlier. Reapply the *Cat Burglar* pruning criteria by loading the *Cat Burglar* agent profile you created earlier (or reentering the values manually if you did not save it). Recall that the pruned tree shows the attacks that are possible for the *Cat Burglar*. If the threat agents chosen for analysis are selected on the basis that they are motivated to harm the system, then it is reasonable to assume that they will carry out any attack within their capability. In other words, **the pruned window shows the probable attacks.**

From within the *Cat Burglar* pruned window, select Analyze > Attack Scenarios to create a list of the attack scenarios available to the *Cat Burglar*. In this rather trivial example there is only one scenario, as shown in **Figure 30**. Clicking on the labels for the various indicator functions will cause the list to be sorted on the selected parameter. (In this simple case there is only one scenario, so there is no point in sorting). **If an impact indicator (such as *Damage Cost*) is used for sorting, then the result is a risk prioritized list of the various attacks and their associated vulnerabilities.**

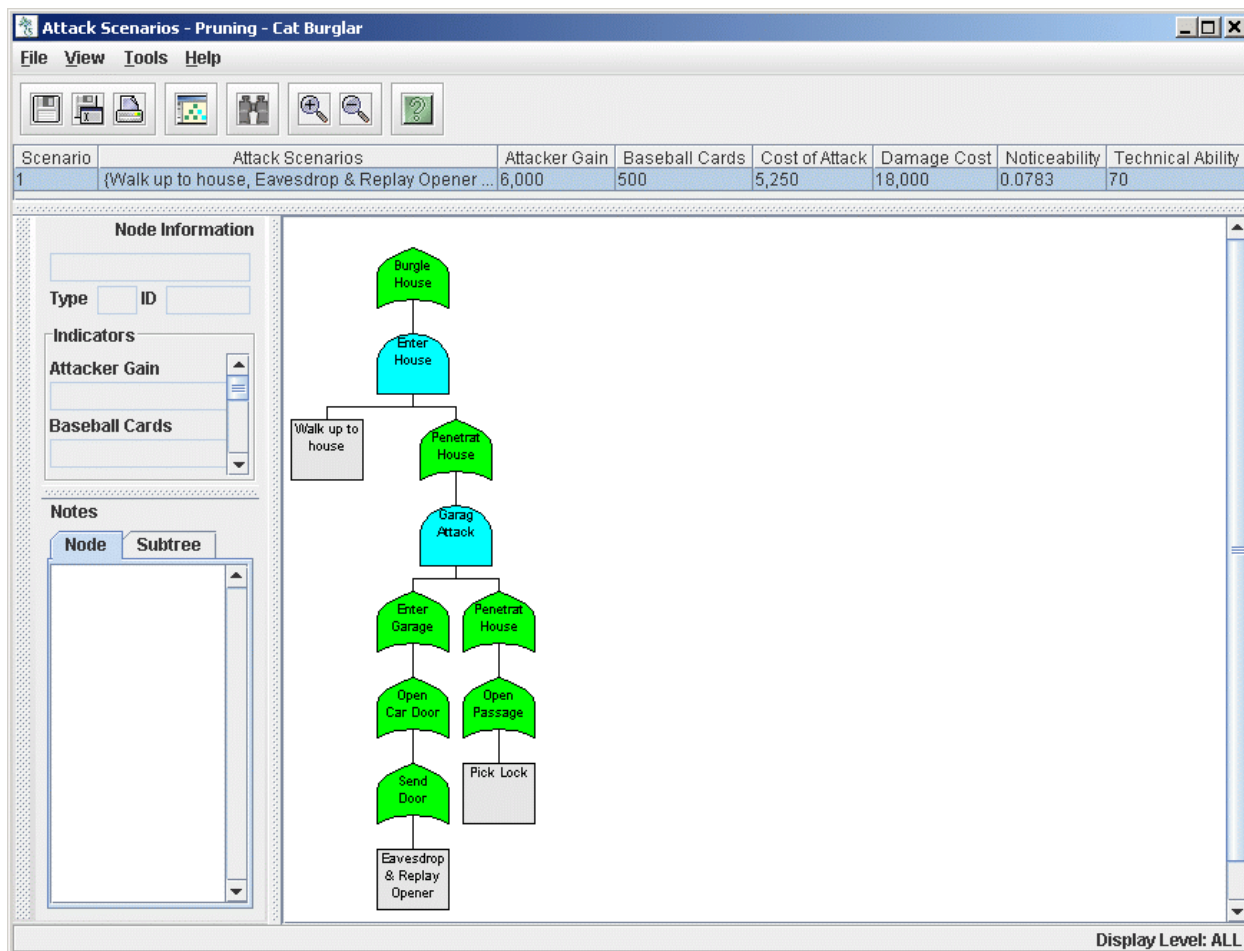


Figure 30

### Advanced Analytic Functionality

The analysis shown above has allowed us to quickly understand what attacks are available to an adversary and to get some idea on the impact on a victim. However, Secur/Tree is capable of much more sophisticated analysis.

As alluded to earlier, whether an attack occurs depends on whether an adversary believes the rewards they will achieve are worth the resources they will expended on the attack (or if they even have them). As with most economic decisions, the value of resources usually depends on their scarcity. Secur/Tree allows an analyst to build a sophisticated model of an adversary's affinity to their resources and to describe in detail the value they place on prospective rewards. This allows a numeric assessment of the likelihood of given attacks to be calculated. Secur/Tree also models the perception of pain that a victim feels given a certain set of consequences.





Together, the estimation of attack probability combined with the perceived impact on the victim allow an accurate determination of risk.

The advanced analytic capability is beyond the scope of this introductory tutorial, but Amenaza Technologies would be pleased to provide you with a demonstration of these features.

## Conclusion

While the house breaking example may have been simple enough (and familiar enough) that you could guess which attacks were available to each kind of attacker, more sophisticated problems are less intuitive. We challenge you to build models of other systems that you are interested in defending.

This tutorial has shown you how graphical attack tree based risk analysis using Secur//Tree can be a powerful tool in the identification of risk. By understanding your adversaries' strengths, weaknesses and the resource requirements to carry out particular attacks, it becomes evident which of your weaknesses are a genuine concern. Threat priority is easy to determine once you can predict how your enemies will behave. Secur//Tree's graphical representations of your systems' risks are the best way to explain risk to everyone, regardless of their job title or position.

In the house break-in example, the attacks available to the two *threat agents* were discussed separately. Secur//Tree can merge these findings by showing the complete collection of attacks from one or more *threat agents*. It can also depict the attacks that selected *threat agents* have in common. These features make it easy to see the "big picture".

Secur//Tree's ability to do "what if" analysis for *threat agents* or countermeasures is very powerful. These thought experiments can be done at any time in the system lifecycle. Although the optimal time to apply Secur//Tree is during the design phase (when changes have the least impact on cost and schedule), it can and should be reapplied whenever changes occur in the environment. Companies with existing security vulnerability scanning tools will be pleased to know that the output from these tools can be used as information sources for Secur//Tree models.

While the house break-in example is simple enough for you to guess which attacks are available to the chosen *threat agents*, complex systems are much less intuitive. If a simple (and arguably incomplete) model of attacks against a residence required over 20 nodes to model, imagine how much more information must be managed to model the information systems in a large company. To model this level of complexity, an automated tool is practically mandatory. No matter how complicated the system, Secur//Tree can quickly and easily model it through the use of capabilities-based, attack tree analysis.

Of course, if every IT problem required that every component be modeled from scratch, the effort could become overwhelming – even with Secur//Tree. To address this valid concern, Amenaza has developed numerous libraries containing predefined attack trees for popular technologies such as Netscape™, Oracle™, Solaris™ and Windows 2000™. You can insert



# Amenaza

TECHNOLOGIES LIMITED

them into your models in much the same way as adding a node. These threat tree libraries are easily customized to match local environments. Existing libraries are updated and additional libraries are added regularly. These updates are available at no charge to customers who have purchased a technical support contract. A sample of the libraries is included with the evaluation copy of Secur/Tree. A more complete set of libraries is included with a purchased copy of the product.

Just as your adversaries use their knowledge about you to attack your systems, Secur/Tree allows you to understand your adversaries' capabilities and forecast where and how they will attack. Armed with this understanding, you can better focus your efforts, your resources and your budget on the most appropriate defenses.



## Secur/Tree® – Dare you risk IT?

*Amenaza Technologies Limited has developed the world's most advanced Attack Tree based vulnerability assessment tool, Secur/Tree®. When used with the accompanying methodology and attack tree libraries, Secur/Tree allows enterprises to discover which weaknesses are most likely to be used against them by attackers. Secur/Tree turns the tables on the attackers by enabling enterprises to quickly and efficiently invest in those security measures that result in the greatest reduction of risk.*

*Learn more about Amenaza Technologies and Secur/Tree at <http://www.amenaza.com>*

*The information and product features described in this document are subject to change without notice. Any discussion of product features or enhancements must not be construed as a commitment by Amenaza Technologies Limited.*