SPARSE: A Hybrid System for Malcode-Bearing Document Detection

Wei-Jen Li
Thesis Defense
June 25, 2008

Advisor: Salvatore J. Stolfo
Computer Science, Columbia University
Outline

- The Threat
- Static Analyses
- Dynamic Tests
- The Hybrid Detection System
- Conclusion
The Threat

- Document formats are defined by complex proprietary structured formats
  - Scripts, tables, media, and arbitrary code
  - Documents can harbor all sorts of malcode aimed at different application objects

- Attack Strategies
  - Macros, vulnerabilities, phishing
  - Stealthy and targeted

- Solutions
  - Disable all macros
  - Patches
  - Tell people not to open email attachments

An Excel chart embedded in a Word document.
Prior Work on Detection Techniques

- Detection techniques
  - Signatures
    - [Kephart 94][Kreibich, 03][Singh, 04][Kim, 04][Newsome, 05]
  - Sandboxes (VMs)
    - [Norman Sandbox, 02][CWSandbox, 07][TTAnalyze, 06]
- More attack strategies
  - Polymorphic
    - [ADMmutate. 2001][Detristan, 03][Kolesnikov, 06] [Perdisci, 06][Song, 07]
  - Mimicry
    - [Kruegel, 05][Wagner, 02]
  - Attacks on VMs
    - [Ferrie, 06]
Problem Statement

- Develop techniques that can identify document files that are poisoned with zero-day malcode
  - Determine whether the documents inspected are benign or malicious
  - Identify the actions and the intent of malcode embedded in documents
  - Extract the embedded malcode from documents to update the detection models
Thesis Hypothesis

By exploiting the vulnerabilities in documents, numerous attack vectors exist that are selective and stealthy; some mimic their surrounding media they are embedded in. Thus, it is difficult to provide accurate detection (i.e. high detection rate and a zero false positive rate) by using a single static model or a dynamic approach. We conjecture a combination of techniques that combine multiple methods and detectors will likely improve detection performance and enhance security of systems.
“…combine multiple methods and detectors will likely improve detection performance and enhance security of systems”

- **Attackers must devise evasion techniques to avoid malcode detection**

- **Attackers have to work far harder and expend more resources than simply executing**
  - “insert $\rightarrow$ object $\rightarrow$ file.malcode”

- *(Quantifying this effort is another PhD thesis…)*
Approach

- SPARSE: A hybrid detection system that integrates both static and dynamic techniques with various modeling and detection strategies
- Documents are analyzed in a sandbox to avoid harming operational systems
Contributions

- **SPARSE**: A fully implemented hybrid detection system to test and evaluate methods to detect malcode-bearing documents.
- A comparative evaluation of three alternative modeling strategies to facilitate detection including a fine-grained parsing mechanism, 2-class learning strategy, and group-specific modeling strategy.
- A complete implementation and exhaustive tests of static statistical n-gram based file binary content analyses that use these modeling strategies tested against thousands of documents.
- *Combination of* a series of dynamic detection methods including environment diversity, a dynamic run-time system event detector, and data randomization.
- A dynamic test scheme to automatically find and extract embedded malcode.
- An automaton that emulates users’ interaction with documents to launch passive embedded objects.
# Experimental Data Sets

<table>
<thead>
<tr>
<th>Category</th>
<th>ARDA</th>
<th><a href="http://www.columbia.edu">www.columbia.edu</a></th>
<th><a href="http://www.nj.gov">www.nj.gov</a></th>
<th>Malware</th>
</tr>
</thead>
<tbody>
<tr>
<td># of docs</td>
<td>1300</td>
<td>1775</td>
<td>3050</td>
<td>2949</td>
</tr>
<tr>
<td>File size</td>
<td>mean: 57kb</td>
<td>mean: 119kb</td>
<td>mean: 178kb</td>
<td>mean: 50kb</td>
</tr>
<tr>
<td></td>
<td>max: 123kb</td>
<td>max: 8000kb</td>
<td>max: 12mb</td>
<td>max: 3mb</td>
</tr>
<tr>
<td></td>
<td>min: 2kb</td>
<td>min: 5kb</td>
<td>min: 11kb</td>
<td>min: 6kb</td>
</tr>
<tr>
<td>Timeframe</td>
<td>2001-01-01 to</td>
<td>1996-12-10 to</td>
<td>1997-12-19 to</td>
<td>1996-02-08 to</td>
</tr>
</tbody>
</table>

29 were provided by ARDA and 1 was a recent attack provided by ITSN.
Outline

- The Threat
- Static Analyses
- Dynamic Tests
- The Hybrid Detection System
- Conclusion
The Static Analyses

Hypothesis:

- Without being executed, the binary content of a particular class of documents can be analyzed by static means to produce probabilistic content models that would reliably identify abnormal embedded data indicating the presence of malware.

- It is necessary to parse/extract the binary document format into its constituent embedded object structure and to model the individual data objects separately.
The Object Linking and Embedding (OLE2) Format

- Embed and link documents and other objects developed by Microsoft

Byte occurrence frequency of the parsed sectors: WordDocument, 1Table, Macros, and Data. The values are computed by using 120 benign documents with macros embedded.
SPARSEGUI
N-gram Based Intrusion Detection Algorithms

- **N-grams:**

  - Shaner’s algorithm [Shaner, 99]
    - Compute a mixture of n-grams, rather than one fixed size n-gram
    - Too expensive
    - Ignore never-before-seen n-grams

- **Anagram [Wang, 06]**
  - Bloom filter
  - Space and time efficient
  - Binary-based analysis

- **Mahalanobis distance [Li, 05][Stolfo, 06], CNG [Abou-Assaleh, 04], and N-Perm [Karim, 05]**
  - Frequency-based analysis
Modeling Strategies

- Using the parser
- 1-class anomaly detection and 2-class mutual information strategy
- Group-specific modeling
- [Stolfo (ARDA report), 06]
Using the Parser

- **Without parsing**
  - Train the model by scanning through the entire training documents, and test a document by using all of its byte content

- **With parsing**
  - Parse/extract the binary document format into “sectors,” one sub-model for each sector
  - When testing, compute a score $S_i$ for each sub-model
  - Final score: $S = \sum_{i=0}^{N} W_i \times S_i$
  - Current models: WordDocument, 1Table, Macros, Data, Others
Evaluation of Using and Not Using the Parser

- Tested 6125 benign and 2949 malicious documents
- C4.5 cross-validation
1-class / 2-class Modeling Strategy

- **1-Class Anomaly Detection:**
  - Compare the score with a predefined threshold

- **2-Class Mutual Information Strategy**
  - Train a “benign” model and a “malicious” model
  - Compare a testing document to both models

\[
\frac{\text{Score}_{\text{benign}}}{\text{Score}_{\text{malicious}}} \Rightarrow \begin{cases} 
> 1 : d \subseteq \text{benign} \\
< 1 : d \subseteq \text{malicious} \\
= 1 : \text{equal}
\end{cases}
\]}
Group-Specific Modeling Strategy

- It is generally hard to train a number of samples to represent “normal”
- How many documents are enough for training?
- Group-specific modeling
  - Conjecture: documents being created, opened and exchanged within a specific group are likely to be similar
  - Train data from a specific corpus to build a model representing “normality” on the site that our IDS is protecting
Evaluation of the Group-Specific Strategy

- Compare 4 groups of document: 3 benign (A, B, C), 1 malicious (D)

- Mahalanobis distance:
  - Documents from the same group are similar (i.e. the lowest distance)

- Different training/testing methods

<table>
<thead>
<tr>
<th></th>
<th>A1</th>
<th>B1</th>
<th>C1</th>
<th>D1</th>
</tr>
</thead>
<tbody>
<tr>
<td>A2</td>
<td>79.42</td>
<td>101.60</td>
<td>79.41</td>
<td>110.83</td>
</tr>
<tr>
<td>B2</td>
<td>82.12</td>
<td>96.93</td>
<td>86.29</td>
<td>103.82</td>
</tr>
<tr>
<td>C2</td>
<td>84.39</td>
<td>107.19</td>
<td>74.07</td>
<td>109.21</td>
</tr>
<tr>
<td>D2</td>
<td>98.08</td>
<td>127.44</td>
<td>102.94</td>
<td>75.54</td>
</tr>
</tbody>
</table>

- Train A1 (benign) and D1 (malicious)
  - Test A2: TP: 99.2%, FP: 0.3%
  - Test B2: TP: 99.2%, FP 3%

- Train both A1 and B1 (benign) And from D1 (malicious)
  - Test A2: TP: 99%, FP 0.3%

- Train only A1 (benign) and D1 (malicious)
  - TP: 99.2%, FP 0.3%
Discussion of the Static Analyses

- The three modeling strategies are effective to ensure the quality of training models and detection performance.
- It is necessary to parse documents and model different types separately.
- Anagram is superior to the other (frequency-based) algorithms that tested in our experiments.
- Short (or small), mimicry, or polymorphic malcode is hard to detect by using statistics-based methods.
- Static analysis cannot identify whether the “intent” of embedded code is benign or malicious.
Outline

- The Threat
- Static Analyses
- Dynamic Tests
- The Hybrid Detection System
- Conclusion
Dynamic Tests

- **Hypothesis:**
  - By executing the documents and inspecting their dynamic run-time system behavior, we will be able to observe the intent of code embedded in documents and to identify stealthy, crafted malcode.

- **Observe the dynamic run-time behavior when opening documents:**
  - System crash, program error, file creation/modification, module loading, registry access/change, process activity/relationship, and thread information.
Dynamic Detection Techniques Used in SPARSE

- Detection methods:
  - Environment diversity
  - Dynamic system event model
  - Automaton
  - Document data randomization

- Malcode locator
  - To identify the location of the embedded malcode
Dynamic Run-time Tests Using Environment Diversity

Observe abnormal behaviors, such as system crash, program corrupt, network connection failure, abnormal macros, memory error, or any other abnormal behavior.
The malcode causes the program crash in CrossOver emulated environment
Monitoring and Modeling the Run-Time System Behavior (Dynamic Behavior Analyzer)

- **Process Monitor**
  - Tree structure: process
  - Model: Anagram
  - Tokens: registry, file, module, thread...

- **AutoIt**
  - Additional pop-up information
Dynamic Behavior Analyzer Experiment

C4.5 cross-validation

1-class

2-class
Automaton

- Launch passive embedded objects for test
- Simulate users’ interaction with documents
- AutoIt (http://www.autoitscript.com): a VB like programming language designed for automating the Windows GUI and general scripting
  - Moving the cursor
  - Clicking buttons
  - Utilizing Hot-keys
  - Saving and closing the document
Launch the Embedded Object
- Used to detect stealthy embedded malcode (i.e. in the 1Table sector, not macros)
- Randomly increase or decrease the data byte values by some arbitrary value \( x \) (e.g., changing the character “A” to “B”)
- After randomizing data value
  - Normal: the display might be distorted, but can be opened without error.
  - Stealthy embedded code: the malcode would be either disabled or crash
Traditions and Cultures 104 - Lecture 851
Winter 2000/2001

Justice and Virtue

Description

This course will introduce you to some of the central questions and ideas in moral and political philosophy through the works of some of the most important figures in the western tradition. We will discuss The Republic by Plato (427 - 347 BCE), The Prince by Niccolo Machiavelli (1469-1527), Leviathan by Thomas Hobbes (1588-1679), Second Treatise on Civil Government and A Letter Concerning Toleration by John Locke (1632-1704), The Social Contract and A Discourse on the Origin and Foundation of Mankind by Jean-Jacques Rousseau (1712-1778), and A Discourse on the Origin and Foundation of Mankind by Jean-Jacques Rousseau (1712-1778).
Embedded Malcode

After randomization: ED 08 91 92 5B 4D 01 48 90 BD 2E FF 80 7E 6B 7E 6B 00 FF D3 EC F8 68 13 11
# Data Randomization Experiment

Attacks in 1Table:

<table>
<thead>
<tr>
<th>Index</th>
<th>Attack type</th>
<th>Detect</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Memory corruption</td>
<td>Yes</td>
<td>FIB</td>
</tr>
<tr>
<td>2</td>
<td>Memory corruption</td>
<td>Yes</td>
<td>fcPilfLfo</td>
</tr>
<tr>
<td>3</td>
<td>Memory corruption</td>
<td>Yes</td>
<td>fcPilfLfo</td>
</tr>
<tr>
<td>4</td>
<td>Display pop-up message</td>
<td>Yes</td>
<td>fcSttbfffn</td>
</tr>
<tr>
<td>5</td>
<td>Encrypted malcode</td>
<td>Yes</td>
<td>fcSttbfffn</td>
</tr>
<tr>
<td>6</td>
<td>Steganographic</td>
<td>Yes</td>
<td>fcSttbfffn</td>
</tr>
<tr>
<td>7</td>
<td>System32/calcalc.exe</td>
<td>Yes</td>
<td>fcPilfLfo</td>
</tr>
<tr>
<td>8</td>
<td>Program error</td>
<td>Yes</td>
<td>fcSttbfffn</td>
</tr>
<tr>
<td>9</td>
<td>Program error</td>
<td>Yes</td>
<td>fcDop</td>
</tr>
<tr>
<td>10</td>
<td>Program error</td>
<td>Yes</td>
<td>fcSttbfffn</td>
</tr>
<tr>
<td>11</td>
<td>Program error</td>
<td>Yes</td>
<td>fcSttbfffn</td>
</tr>
<tr>
<td>12</td>
<td>Trojan-Dropper</td>
<td>No</td>
<td>Multiple (5)</td>
</tr>
<tr>
<td>13</td>
<td>Trojan-Dropper</td>
<td>No</td>
<td>Multiple (25)</td>
</tr>
<tr>
<td>14</td>
<td>Trojan-Dropper</td>
<td>No</td>
<td>Multiple (22)</td>
</tr>
<tr>
<td>15</td>
<td>Trojan-Dropper (an attack Used to against ITSN in March 2008)</td>
<td>No</td>
<td>fcSttbfBkmkFactoid</td>
</tr>
</tbody>
</table>

- For the 3 Trojan-Droppers, the malicious “behaviors” were disabled. They were considered not detected because they didn’t crash.

- The structure of the last attack is an undocumented structure, which we didn’t randomize it’s value

- Test 1516 benign documents: 9 FPs (0.59%)
Malcode Locator

- Used to find the malicious portions of a document that has been detected to harbor malcode
- First, identify the malicious (anomalous) dynamic behavior
- Remove the embedded structures one by one and verify the dynamic behavior
Discussion on the Dynamic Tests

By using the dynamic tests, we can observe the behavior of code

- **False positives**
  - Some documents downloaded from legitimate websites cannot be opened
  - Some benign documents create unusual registry keys, templates, or files

- **False negatives**
  - Stealthy malcode that does not generate sufficient amount of abnormal events
Outline

- The Threat
- Static Analyses
- Dynamic Tests
- The Hybrid Detection System
- Conclusion
Control Flow

Document

Static Detector:
Detects malicious file binary content

If labeled positive
Malcode Locator:
Tests the new generated documents in which each has a portion removed

If labeled positive
1. Report an attack
2. Update the malicious model of the static detector
3. Update the malicious model of the dynamic detector

If labeled negative

Dynamic Detector:
Detects malicious system behavior

If labeled negative
The document is benign

Document Parser

If labeled negative
The document is benign
Evaluation of the Hybrid System

- Incremental test (test zero-day)
- Update strategies
  - 1: update the model if the static detector “OR” the dynamic detector says positive
  - 2: update the model if the static detector “AND” the dynamic detector says positive
Discussion of the Hybrid System

- False positives
  - Some benign documents cannot be opened (true positive?)
  - Create unusual registry keys, templates, or files

- Possible attacks
  - Logic bombs
  - Stealthy attacks
Contributions

- SPARSE: A fully implemented hybrid detection system to test and evaluate methods to detect malcode-bearing documents.
- A comparative evaluation of three alternative modeling strategies to facilitate detection including a fine-grained parsing mechanism, 2-class learning strategy, and group-specific modeling strategy.
- A complete implementation and exhaustive tests of static statistical n-gram based file binary content analyses that use these modeling strategies tested against thousands of documents.
- Combination of a series of dynamic detection methods including environment diversity, a dynamic run-time system event detector, and data randomization.
- A dynamic test scheme to automatically find and extract embedded malcode.
- An automaton that emulates users’ interaction with documents to launch passive embedded objects.
Conclusion

- The hybrid detection system, SPARSE, combined with various detection strategies can enhance security of document processing.
- Attackers have to expend more effort to fashion embedded malcode and evade detection.
Future Work

- Improve the automaton such as interacting with web browsers
- Improve the virtual machine emulator so logic attacks may be observed
- Convert the display text to a harmless image form to remove embedded malcode (OCR)
- Apply the same integrated detection technique to inspect malicious PDF documents
- Collaboration of various detectors among a large number of sites
The End
Backup Slides
Object Lining and Embedding (OLE2) Format

<table>
<thead>
<tr>
<th>Section</th>
<th>Sizes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>(512)</td>
</tr>
<tr>
<td>Big Blocks</td>
<td>(Multiple of 512)</td>
</tr>
<tr>
<td>Properties</td>
<td>(Multiple of 128)</td>
</tr>
<tr>
<td>Small Blocks</td>
<td>(Multiple of 128)</td>
</tr>
</tbody>
</table>

Byte occurrence frequency of the parsed sections: WordDocument, 1Table, Macros, and Data. The values are computed by using 120 benign documents with macros embedded.
Anagram Algorithm

- Bloom filter: a space and time efficient hash table
Anagram Algorithm

- Train a model by using known exemplar files

- Test an unknown file against the model, and compute a “similarity” score:

  \[
  \text{Score} = \frac{N_{\text{seen}}}{T} = 1 - \frac{N_{\text{new}}}{T} \in [0,1]
  \]

- Only checks whether an n-gram exists in the model – does not consider frequency
Static Experimental Results

<table>
<thead>
<tr>
<th></th>
<th>Mahalanobis Distance</th>
<th>Anagram</th>
<th>CNG</th>
<th>N-Perm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Without Parsing</td>
<td>With Parsing</td>
<td>Without Parsing</td>
<td>With Parsing</td>
</tr>
<tr>
<td>TP</td>
<td>98.8%</td>
<td>96.88%</td>
<td>98.96%</td>
<td>95.56%</td>
</tr>
<tr>
<td>FP</td>
<td>47.12%</td>
<td>19.25%</td>
<td>10.3%</td>
<td>0.99%</td>
</tr>
<tr>
<td>Overall Accuracy</td>
<td>79.9%</td>
<td>90.27%</td>
<td>95.15%</td>
<td>97.47%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Additional Tests

Different Training Size: 5-gram

Performance of different n-grams

Using different number of training documents

Using different length of n-grams
Bloom filter: used bits of different document sectors

---

**Bloomfilter Used Bits**

- **WordDocument**
- **Data**

**Bloomfilter Used Bits**

- **1Table**
- **Macros**
- **Others**
Number of Occurred Distinct N-Grams

X: number of documents read
Y: number of distinct n-grams
The Likelihood of Seeing New N-Grams in Benign Documents
Shaner’s Patent

- Identify the type of an unknown file
- Train a set of file types using a set of exemplar files for each type
- A distribution of a mixture of n-grams (1~8)
- Too expensive: $256^n$
- Ignore never-before-seen n-grams

\[ F(i) = \log(K(x_i / N_t) + 1) \]
\[ W_t(i) = F(i) \ast U(i) \ast A^d(i) \]
\[ U(i) = (1/\log(N_t)) \sum_j (p_{ij} \log(1/ p_{ij})) \]
\[ \text{Score}_t = \sum W_t(i) \]
\[ A(i) = 1 - ((1/\log L) \sum L p_i \log(1/ p_i)) \]
Other Static Methods

- Byte Value Entropy Analysis
- File Content Differences Identify Embedded Malcode
Entropy Analysis
Embedded Malcode
Compare to a COTS AV Scanner

- Test 35 documents where the ground truth was unknown before the test
- 10 benign and 25 malicious (17 malicious macros, 8 Table exploits)

<table>
<thead>
<tr>
<th></th>
<th>Static detector</th>
<th>Dynamic detector</th>
<th>COTS AV scanner</th>
</tr>
</thead>
<tbody>
<tr>
<td>TP / FN</td>
<td>18 / 7</td>
<td>25 / 0</td>
<td>17 / 8</td>
</tr>
<tr>
<td>FP / TN</td>
<td>0 / 10</td>
<td>0 / 10</td>
<td>0 / 10</td>
</tr>
<tr>
<td>Total Correct</td>
<td>28</td>
<td>35</td>
<td>27</td>
</tr>
</tbody>
</table>
The System Design

Static Detector
- Whitelist
- Blacklist

Dynamic Behavior Analyzer
- Dynamic Behavior Model
- Blacklist

Host Manager

Virtual Network

Dynamic Detector
(VM) Dynamic Emulator

Virtual Network

Malcode Locator
(VM) Malcode Locator

Document Parser
Related Work: Signature Generation

- IBM [Kephart, 94]
- EarlyBird [Singh, 04]
  - Use prevalent invariant strings as signatures
- Honeycomb [Kreibich, 03]
  - Longest common sequences
- Autograph [Kim, 04]
  - Most frequently occurring byte sequences
- Distributed detectors [Goel, 03]
Related Work: N-Gram

- MEF [Schultz, 01]
- Fileprints [McDaniel, 03]
- CNG [Abou-Assaleh, 04]
  - Most frequent n-grams
  - Only read a portion of the file
    \[
    \sum_{s \in \text{profiles}} \left((f_1(s) - f_2(s))/(\frac{f_1(s) + f_2(s)}{2})\right)^2
    \]
- Mahalanobis distance [Li, 05]
  \[
  D(x, \bar{y}) = \sum_{i=1}^{n} (|x_i - \bar{y}|/(\sigma_i + \alpha))
  \]
- N-Perm [Karim, 05]
Related Work: Dynamic

- System call
  - First work [Forrest, 96],
  - Longest mismatched sequences [Wespi, 03]
  - Finite State Automaton [Sekar, 01]
  - System call arguments [Kruegel, 03][Tandon, 03][Bhatkar, 06]
  - System call stack [Feng, 03][Gao, 04]

- RAD [Apap, 03] [Sidiroglou, 03]

- Taintcheck [Newsome, 05], Panorama [Yin, 07]
Related Work: Dynamic

- Sandbox
  - Norman Sandbox [Natvig, 02]
  - TTAnalyze [Bellard, 05]
  - CW Sandbox [Willems, 07]
  - Strider HoneyMonkeys [Wang, 06], BrowserShield [Reis, 06]
Related Work: Advanced Attacks

- Steganalysis [Provos, 02]
- Polymorphic
  - [Szor, 01], ADMutate, CLET (cramming) [Detristan, 03]
  - Georgia Tech [Kolesnikov, 06], Noise Injection [Perdisci, 06], Paragraph (spurious features) [Newsome, 06], [Song, 07]
- Mimicry attacks
  - no-op, equivalences [Wagner, 02]
  - [Kruegel, 05]
  - Control-flow interposition [Parampalli, 08]
Related Work: Advanced Attacks

- Code injection
  - Protect memory: StackGuard PointGuard [Cowan, 98, 03] Libsafe [Baratloo, 00], PAS: executable space protection [Busser, 04]
  - ASLR [Busser, 04] [Bhatkar, 03], attack ASLR [Shacham, 04]
  - ISR [Kc, 03] [Barrantes, 03], attack ISR [Sovarell, 05]
Accomplishment

Some pop-up information
Some pop-up information
When you move the mouse into the above text field, a Javascript will try to fetch C:\boot.ini from your computer, and instruct your browser to go to http://shh.thathost.com/secdemo/show.php with the content of the file as a parameter.

In other words: The script reads a file from the local computer, and sends it off to a remote site.

At least this works on my recently updated Windows XP box, with a recently updated...
This file appears to use a new format that this version of Acrobat does not support. It may not open or display correctly. Adobe recommends that you upgrade to the latest version of our Acrobat products. Please visit our product site at http://www.adobe.comacrobat

Do not show this message again

OK

There was an error opening this document. The file is damaged and could not be repaired.

OK

blur

OK
The page cannot be displayed

The page you are looking for is currently unavailable. The Web site might be experiencing technical difficulties, or you may need to adjust your browser settings.

Please try the following:

- Click the Refresh button, or try again later.
- If you typed the page address in the Address bar, make sure that it is spelled correctly.
- To check your connection settings, click the Tools menu, and then click Internet Options. On the Connections tab, click Settings. The settings should match those provided by your local area network (LAN) administrator or Internet service provider (ISP).
- See if your Internet connection settings are being detected. You can set Microsoft Windows to examine your network and automatically discover network connection settings (if your network administrator has enabled this setting).
  1. Click the Tools menu, and then click Internet Options.
  2. On the Connections tab, click LAN Settings.
  3. Select Automatically detect settings, and then...