Agenda

■ The challenge of business logic bots
  ▸ Business logic attacks
  ▸ Business process automation:
    ▪ The friendly side of web automation
  ▸ Business logic bots:
    ▪ Malicious web automation

■ Solutions
  ▸ Detection
  ▸ Mitigation
## Business Logic Attacks (BATs)

Compared to syntactic attacks:

<table>
<thead>
<tr>
<th>Technical Attacks</th>
<th>Business Logic Attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Malformed requests</td>
<td>Normal requests</td>
</tr>
<tr>
<td>Invalid input values</td>
<td>Legitimate input values</td>
</tr>
<tr>
<td>Change functionality</td>
<td>Abuse functionality</td>
</tr>
<tr>
<td>Attack the application and only indirectly the business</td>
<td>Attack directly the business</td>
</tr>
<tr>
<td>Usually a single request</td>
<td>Often multiple requests</td>
</tr>
</tbody>
</table>
Web Automation
Web Automation

- The fact is that web automation is in wide use
  - Online form automation
  - Tracking competition
  - Personal and institutional stock trading
  - Indexing services
  - Comparative shopping
  - Web Services and other web APIs

- Bottom line is that business level automation may or may not be defined as an attack based on the context of things
  - Who is the source
  - Which part of the business logic is being invoked
Born to be bad:

**BUSINESS LOGIC BOTS**

(BlBs)
What BLBs Are Used For

- **Brute force**
  - Cracking login credentials
  - Guessing session identifiers, file and directory names

- **Denial of Service**
  - Locking resources
  - Abusing resource-sensitive functions

- **Web Spam**
  - Abusive SEO
  - Comment Spam

- **Click Fraud**
  - Referrer click fraud.
  - CSRF click fraud
Hardcore Robotics

■ Queue Jumping
  ▸ Ticketmaster confessed to “fighting like the dickens” queue jumping.
  ▸ Travel agents known to automate air line ticketing systems.

■ Auctions Sniping
  ▸ Watching a timed online auction and placing a winning bid at the last possible moment giving the other bidders no time to outbid the sniper.

■ Poll Skewing
Gaming Bots

■ MUD, Virtual Worlds & Second Life bots:
  - Gain Wealth, and turn it into money in Second Life.
  - Scripted Clients
  - GUI Bots

■ Poker Bots:
  - Share information between several bots at one table.
  - Monitor tables to choose the weak ones.
  - Play well.
Information Harvesting

- Harvests:
  - E-mail and personal information
  - Competitive information
  - Record oriented information such as CVs
  - Entire Web sites for creating a mirror

- Executed from:
  - Local computer
  - Distributed, potentially using bot net
  - Trojans, exploiting the victims credentials at the site
ENOUGH WITH THE FUD!
Solutions

- The solution is comprised of two separate problems
  - Detection
  - Mitigation

- Detection
  - Detect automation (absolute)
  - Flag unauthorized use of automation (subjective)

- Mitigation
  - Effective
  - Does not break application
Detection – Basic Tools

- Black listing:
  - IP Addresses (IP Reputation) – Anonymous Proxies, TOR exit nodes, highly active bots
  - User Agents
  - Ad-hoc attack vector patterns
  - Ad-hoc comment spam patterns

- Request structure
  - Missing / mismatch Host header
  - Irregular header combinations

- Naïve, but eliminates the masses
Detection – Proactive Techniques

- Introduce extra content into the response
  - The extra content is interpreted in a different manner by a human driven browser and by an automated tool
  - Must not affect visuals
  - Must not break application

- Positive detection
  - Extra content affects a robot but not human

- Negative Detection
  - Extra content affects a browser but not a robot
Detection – Frequency Measurement

- Count the frequency of “events” within some scope in a given time frame

- Challenges
  - What’s an event?
  - What’s the best scope?
  - What’s the right threshold?

- Allow detection of script related attacks and brute force attacks
Detection – Flow

- Some attacks, either inherently or for performance reasons bypass normal application flow
  - Traversing a product catalog
  - Skipping transaction validation
- Not easy to implement
  - Referer header can be forged
  - Flows are hard to define and track in modern applications that use frames and AJAX
- Require guided configuration and learning algorithms
- Can detect some types of forceful browsing and man in the browser attacks
Detection – Click Rate

- Humans take time to respond (even the fast ones)
- Some observations:
  - Clickable events, within a session, need to be at some minimal distance from one another
  - Within a session, over time, clickable events should be relatively slow
- Can detect general script attacks as well as man in the browser attacks
Detection - Summary

- Will a single method do the trick?
  - I don’t think so.
- Will there be false positives?
  - Yes!
- Do I care?
  - No! Let me tell you why...
Mitigation

- Attacks are automated
  - I can’t prevent the attack from going on
  - I can however try to “defuse” its effects
  - Examples:
    - Slow down a brute force attack
    - Reduce the rate of a DDoS attack
    - Make the victim aware of a man-in-the-browser attack
    - Enforce flow on transactions
    - Disinformation
  - Preventative measures may increase the cost of automation to the level that makes it much less attractive for anything but high end targets
Mitigation - Blocking

- Dropping requests can only occur in very specific cases
  - IP blacklists
  - User-agent blacklists
  - Strongly enforced flow (e.g. through nonce in a form)

- Dropping requests that fail to answer the challenges described in the following slides
Mitigation – That Which Makes Us Human

- Provide a Turing test that only a human can solve.
- Usually called CAPTCHA. Traditionally character recognition
- Other methods exist
  - Choose the correct description of an image
  - Solve a simple riddle

John had one thousand apples and five oranges. He ate as many of his apples as there is letters in word "apple". Also he ate two bananas :-). How many apples does John have?
Mitigation – That Which Makes Us Human

- There are automated tools and algorithms today that solve CAPTCHA’s of various types.
- I don’t care.
  - If a brute force login program solves one CAPTCHA per second then it is ineffective.
  - If a client solves a CAPTCHA faster than a human being (no less than one second) then it can easily be identified as a robot and further challenged (see next slide).
Mitigation – Throttling

- Slowing down an attack is most often the best way to make it ineffective.
  - A second of delay can make the difference for an automated attack but will not be noticed by most humans
- Server side throttling may have severe impact on server (quickly consume connection resources)
- Client side computational challenges
  - Client is required to solve a computational challenge that can be easily verified by server
  - Code for solving the challenge is introduced into the response in the form of a script
Mitigation – Adaptive Authentication

- When automation is detected in the context of a user (man in the browser)
- Ask for additional authentication
  - Repeat password
  - Previously recorded questions
- Makes the attack apparent to a victim
Mitigation - Disinformation

- Feed the client with bogus information
- A client follows a hidden link
  - Respond to the request with a page that includes a large number of server distinguishable random links
  - Whenever one of the random links is requested generate yet another random page
- A client that follows a hidden link that was generated by a script
  - Respond with a page that includes a script that runs for a long time before generating a new random link
Mitigation - Summary

- Mitigation methods should take into consideration the possibility of false positives.
- Most often the system’s reaction to a suspected automation attempt should not be blocking but rather challenging the client:
  - Legitimate clients are not materially affected.
  - Automated clients become ineffective.
Summary

- Automated business layer attacks are proliferating today and expected to grow in number and sophistication in the near term.
- Detecting and mitigating these attacks require a set of sophisticated tools that are different than the standard web application security tools.
- Some of the issues have nothing to do with the way the application code is written.
- It’s bound to be a cat and mouse game as robots become more sophisticated.
- As a consequence of the above, solutions should be external to the application code.
Noa Bar-Yosef, noa@imperva.com