Kill-Bots: Surviving DDoS Attacks That Mimic Legitimate Browsing

Srikanth Kandula

Dina Katabi, Matthias Jacob, and Arthur Berger
CyberSlam = DDoS that Mimics Legitimate Browsing
CyberSlam

20,000+ zombies issue requests that mimic legitimate browsing

GET File.zip

DO DBQuery

www.foo.com

Requests Look Legitimate ⇒ Standard filters don’t help
CyberSlam Attacks Happen!

• Instances of CyberSlam
  □ First FBI DDoS Case - Hired professionals hit competitor
  □ Mafia extorts online gaming sites ...
  □ Code RED Worm

• Why CyberSlam?
  □ Avoid detection by NIDS & firewalls
  □ High pay-off by targeting expensive resources
    • E.g., CPU, DB, Disk, processes, sockets
  □ Large botnets are available
Threat Model

• In scope
  □ Attacks on higher layer bottlenecks, e.g., CPU, Memory, Database, Disk, processes, ...
  □ Attacks that fool the server to congest its uplink bandwidth
  □ Mutating attacks

• Outside the scope
  □ Flooding server’s downlink (prior work)
  □ Live-lock in the device driver
Tentative Solutions

- Filter big resource consumers?
  → No big consumers; Commodity OS do not support fine-grained resource accounting
- Passwords?
  → Might not exist, expensive to check
- Computational puzzles?
  → Computation is abundant in a botnet
Partial Solution:

Reverse Turing Test (e.g., CAPTCHAs) to distinguish humans from zombies

Our website is experiencing unusually high load. To restrict automated access we require code verification. Please enter the code shown in the image below:

![Code verification image]

But...
3 Problems with CAPTCHA Authentication

- (1) DDoS the authentication mechanism (connect to server, force context-switches, hog sockets etc.)

- (2) Bias against users who can’t or won’t answer CAPTCHAs

- (3) How to divide resources between service and authentication as to maximize system goodput?
Kill-Bots' Contributions

- First to protect against CyberSlam
- Solves problems with CAPTCHAs:
  - Cheap stateless authentication
  - Serves legit. users who don’t answer CAPTCHAs
  - Optimal balance between authentication & service
- Improves performance during Flash Crowds
- Order of magnitude improvement in goodput & response time
Kill-Bots is a **kernel extension** for web servers.

![Diagram with nodes and arrows](image)

- **Normal** to **Suspected Attack** with condition: \( \text{LOAD} > L_1 \)
- **Suspected Attack** to **Normal** with condition: \( \text{LOAD} < L_2 < L_1 \)

- **No Overhead**
- **New Clients are authenticated once and given HTTP Cookie**
Problem 1: Authentication vulnerable to DDoS
Problem 1: Authentication vulnerable to DDoS

Client

| SYN  | SYNACK | SYNACKACK | HTTP Request | Send CAPTCHA | TCP FIN |

Server

SYN Cookie

Check cookie, socket, reserve buffers
Causes context switch, buffer copies

Resources are reserved till client
sends a FIN but zombies don’t FIN
**Problem 1:** Authentication vulnerable to DDoS

**Solution:** Modify network stack to issue CAPTCHAs without state
Problem 1: Authentication vulnerable to DDoS
Solution: Modify network stack to issue CAPTCHAs without state

Modified Network Stack
- Stateless & Cheap
- Keep congestion control semantics
- No browser mods.

Drop; Check cookie, send CAPTCHA without a socket!
Problem 2: Legit. Users who don’t answer CAPTCHA
Solution: Use reaction to CAPTCHA

Humans
(1) Answer CAPTCHA
(2) Reload; if doesn’t work, give up

Zombies
Can’t answer CAPTCHA, but have to bombard the server with requests

- Count the unanswered CAPTCHAAs per IP, and drop if more than $T$; Cheap with a Bloom Filter
Stage 1:
- CAPTCHA Authentication
- Learn IP addresses of zombies using Bloom filter

Bloom Learns All Zombie IPs

Stage 2:
- Use only Bloom filter for Authentication
- No CAPTCHAs

Users who don't answer CAPTCHAs can access the server despite the attack in Stage 2
Problem 3: To Authenticate or To Serve?
Problem 3: To Authenticate or To Serve?

- Authenticate all new arrivals
  → can’t serve all authenticated clients
- Authenticate very few arrivals
  → too few legitimate users are authenticated

Solution:

- Authenticate new clients with prob. $\alpha$ (drop others)
  → A form of admission control with 2 arrival types

But what $\alpha$ maximizes goodput?
Analysis

Modeled system using Queuing Theory

Found Optimal $\alpha^*$ (proof in paper)

But $\alpha^*$ depends on many unknown parameters

- attack rate
- mean service time
- mean session size
- legitimate request rate, etc...
Solution to Problem 3:

Kill-Bots adapts the authentication prob. by measuring fraction of time CPU is idle
Solution to Problem 3:

**Kill-Bots adapts the authentication prob. by measuring fraction of time CPU is idle**

- Analysis says: if idle > 0, $\alpha$ is prop. to (1- idle)
- Say you want to keep server busy 90% of time:

\[
\frac{\alpha_{90\%}}{\alpha_{current}} = \frac{0.9}{1 - \text{idle}_{current}}
\]
Solution to Problem 3:

**Kill-Bots adapts the authentication prob. by measuring fraction of time CPU is idle**

- Analysis says: if idle > 0, \( \alpha \) is prop. to \((1 - \text{idle})\)
- Say you want to keep server busy 90% of time:

\[
\frac{\alpha_{90\%}}{\alpha_{\text{current}}} = \frac{0.9}{1 - \text{idle}_{\text{current}}}
\]

- Kill-Bots adapts in real time

\[
\alpha_{90\%} - \alpha_{\text{current}} = \alpha_{\text{current}} \left( \frac{\text{idle}_{\text{current}} - 0.1}{1 - \text{idle}_{\text{current}}} \right)
\]
Solution to Problem 3:

**Kill-Bots adapts the authentication prob. by measuring fraction of time CPU is idle**

- Analysis says: if idle > 0, $\alpha$ is prop. to (1 - idle)
- Say you want to keep server busy 90% of time:

\[
\frac{\alpha_{90\%}}{\alpha_{\text{current}}} = \frac{0.9}{1 - \text{idle}_{\text{current}}}
\]

- Kill-Bots adapts in real time

\[
\Delta \alpha = \frac{1}{8} \alpha_{\text{current}} \left( \frac{\text{idle}_{\text{current}} - 0.1}{1 - \text{idle}_{\text{current}}} \right)
\]
Tying it Together

New TCP

Bloom

no

yes

drop
Tying it Together

Diagram:
- New TCP
- Bloom: no -> Authenticated User?: no -> no
  - yes: drop
  - yes: Serve Req
Tying it Together

New TCP

Bloom: no → Authenticated User?: no → Correct Answer?: no

- yes: drop
- yes: Serve Req
- yes: Dec. Bloom
  - HTTP Cookie
Tying it Together

New TCP

Bloom
    no → Authenticated User?
    yes → drop

Authenticated User?
    no → Correct Answer?
    yes → Serve Req

Correct Answer?
    no → Authenticate? (α)
    yes → Dec. Bloom

Authenticate? (α)
    yes → HTTP Cookie
    no → drop
Tying it Together

New TCP

Bloom

Authenticated User?

Correct Answer?

Authenticate? α

Stage1?

Serve Req & HTTP Cookie

yes

no

yes

no

yes

no

drop

Serve Req

Dec. Bloom

HTTP Cookie

drop

Inc. Bloom

CAPTCHA
Recap: Kill-Bots addresses CyberSlam

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>• DDoS the authentication</td>
<td>• Send CAPTCHAs cheaply without sockets</td>
</tr>
<tr>
<td>• Serve legitimate users who don’t answer CAPTCHAs</td>
<td>• Use reaction to CAPTCHA to identify zombies</td>
</tr>
<tr>
<td>• Divide resources between authentication &amp; service</td>
<td>• Adaptive authentication as admission control</td>
</tr>
</tbody>
</table>
Attacks & Defenses

• Replay Attacks?
  □ Don’t work. Limit #connections per cookie

• Spoof IP, cause Bloom filter to block
  □ Doesn’t happen. SYN cookie before updating Bloom

• Breaking the CAPTCHA?
  □ Kill-bots can use any Reverse Turing Test
Performance
Wide-area Evaluation Using PlanetLab

- Legit. users are driven from CSAIL Web traces
- >25,000 attackers on PlanetLab request random pages
- 60% of legitimate users answer CAPTCHAs
Metrics

- Goodput (of Legitimate Users)
- Response time (of Legitimate Users)
- Maximum survivable attack rate
Kill-Bots under DDoS

Goodput of Legit (Mb/s) vs. Attack Rate (Request/sec)

- **Kill-Bots**
- **Base**

Base Server Crashes
Kill-Bots under DDoS

Goodput of Legit. (Mb/s)

Response Time (sec)

Attack Rate (Request/sec)
5-10 times better Goodput and Response Time

![Graph showing Goodput and Response Time](image)

- **Goodput of Legit. (Mb/s)**
- **Attack Rate (Request/sec)**
- **Response Time (sec)**

Graphs show the performance of Kill-Bots compared to Base under varying attack rates.

- **Base Server Crashes**

The graphs illustrate a significant improvement in Goodput and Response Time for Kill-Bots over the Base scenario.
Why Adapt the Authentication Probability?

Adaptive $\alpha$ is much better than authenticating every new user.
Kill-Bots under Flash Crowd

- Goodput of legit. (Mb/s)
  - Flash Crowd

- Response Time (sec)
  - Base
  - Kill-Bots

Time (sec)
Orders of magnitude better Response Time

![Graph showing goodput and response time over time]

- Goodput of legit. (Mb/s)
- Response Time (sec)
- Flash Crowd
- Base
- Kill-Bots
Adaptive $\alpha$ provides admission control
## Kill-Bots under Flash Crowd

<table>
<thead>
<tr>
<th></th>
<th>Base Server</th>
<th>Kill-Bots</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of dropped</td>
<td>360,000</td>
<td>80,000</td>
</tr>
<tr>
<td>legitimate requests</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**Kill-Bots authenticates new clients only if it can serve them...**
Kill-Bots’ Contributions

• First to protect Web servers from DDoS attacks that mimic legitimate browsing

• First to deal with CAPTCHA’s bias against legitimates users who don’t solve them

• Sends CAPTCHA and checks answer without any server state

• Addresses both DDoS attacks and Flash Crowds

• Orders of magnitude better response time, goodput, and survivable attack rate