

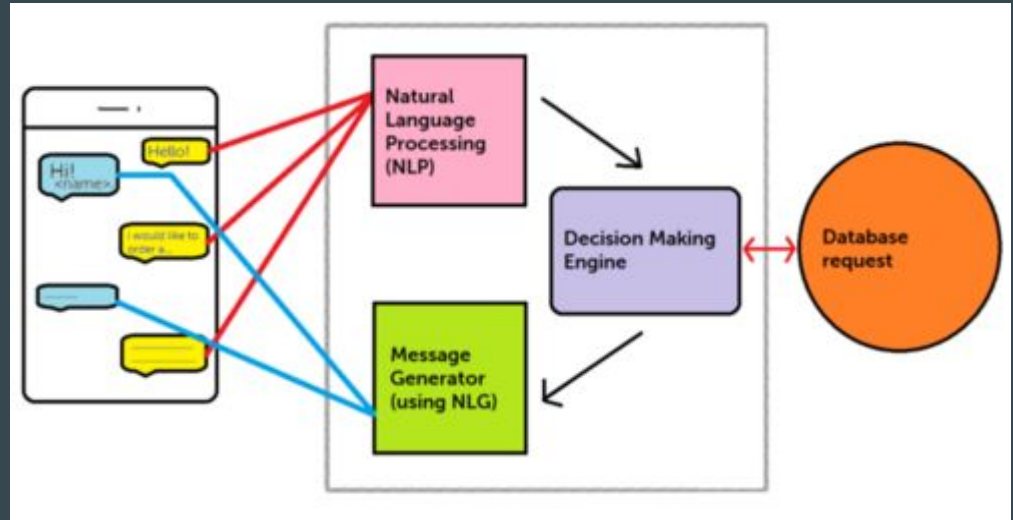
Chatbot Security and Privacy in the Age of Personal Assistants



By Winson Ye and Qun Li
College of William and Mary

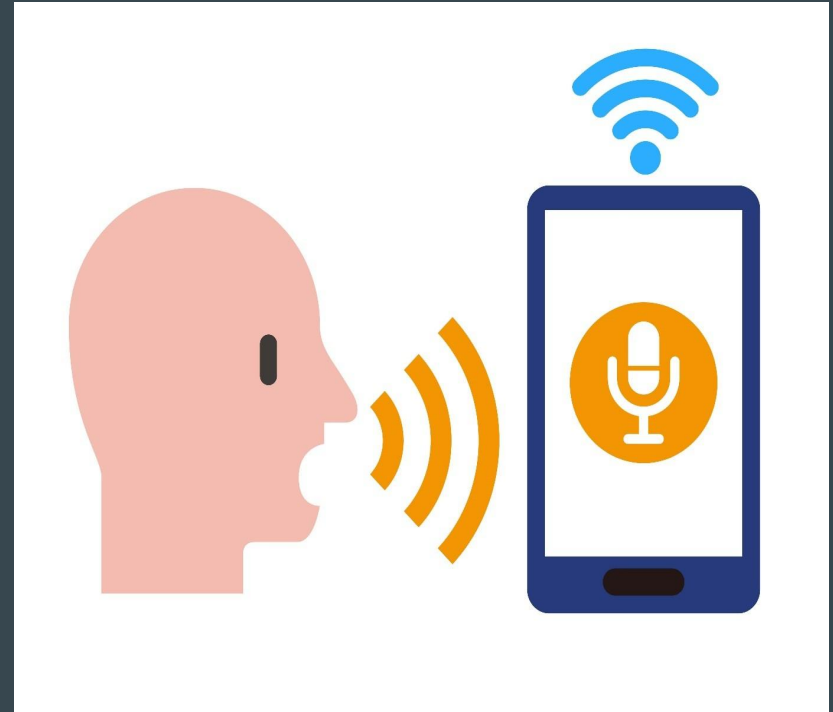
Introduction

- A. Client Module
- B. Communication Module
- C. Response Generation Module
- D. Database Module



Client Module: Unintended Activation Attacks

- Wake-up phrases are used to activate the smart device
 - What if the device confuses words?
 - What if the device records other people in the same room?
 - What if the device is tricked by a recording?
- What are some solutions?
 - Use Wifi to detect human motion
 - Detect whether user is talking to human or device



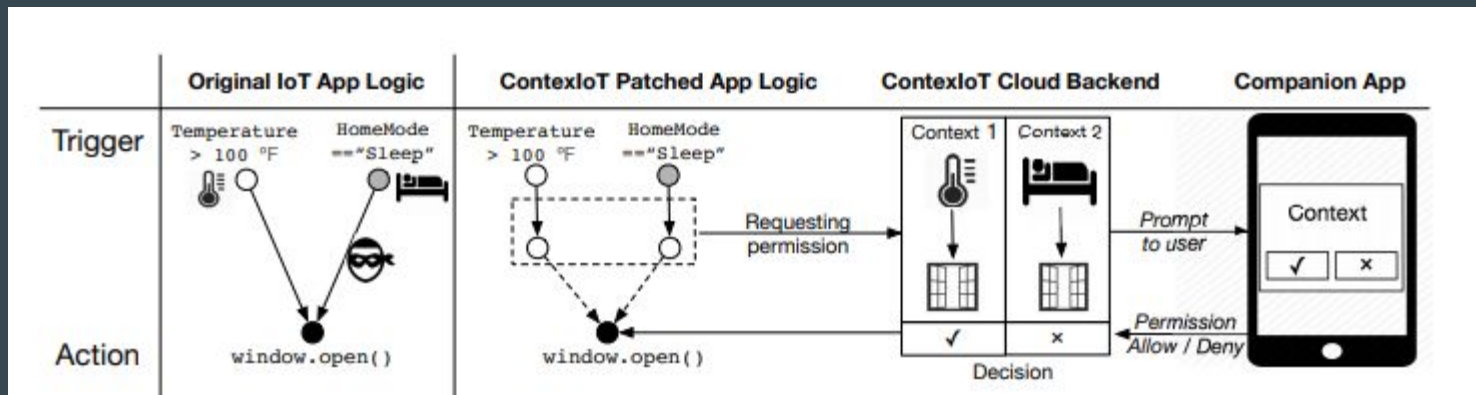
Client Module: Faked Response

- User misconceptions abound
 - 30% of users have trouble turning off smart device
 - 78% did not use LED to check for proper termination
- What if a malicious skill tricks the user into thinking they switched to a different app?
- What if a malicious skill fakes termination?
- What are some solutions?
 - Check smart device responses against a black list



Client Module: Access Control Attacks

- Some apps may grant very broad permissions to the user
 - What if a hacker can take advantage of this to break into the house?
- What are some possible solutions?
 - Defensive coding strategies
 - Security profilers



Client Module: Adversarial Voice Samples

- Voice recognition technology is essential for personal assistants
 - What if we can perturb the voice command such that the personal assistant misinterprets it?
 - What if we can hide voice commands in songs?
- What are some possible solutions?
 - Retrain the model
 - Keep the architecture secret

Communication Module

- DDoS Attacks
 - Flood the server with as many requests as possible
- Wiretapping
 - Use packet metadata to predict voice command
- MitM Attacks
 - Intercept messages and delete/modify them

The screenshot shows a Wireshark interface with a packet capture of a DNS query and response. The packet list pane shows several packets, with packet 349 selected. The packet details pane shows the structure of the DNS response, including the transaction ID, flags, and the answer section which contains the IP address for the domain 'cdn-0.nflximg.com'.

```
tv-netflix-problems-2011-07-06.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/> Expression... +
No. Time Source Destination Protocol Length Info
343 65.142415 192.168.0.21 174.129.249.228 TCP 66 40555 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=491519346 TSecr=551811827
344 65.142715 192.168.0.21 174.129.249.228 HTTP 253 GET /clients/netflix/flash/application.swf?flash_version=flash_lite_2.1&v=1.5&n...
345 65.230738 174.129.249.228 192.168.0.21 TCP 66 80 → 40555 [ACK] Seq=1 Ack=188 Win=6864 Len=0 TSval=551811850 TSecr=491519347
346 65.240742 174.129.249.228 192.168.0.21 HTTP 828 HTTP/1.1 302 Moved Temporarily
347 65.241592 192.168.0.21 174.129.249.228 TCP 66 40555 → 80 [ACK] Seq=188 Ack=763 Win=7424 Len=0 TSval=491519446 TSecr=551811852
348 65.242532 192.168.0.21 192.168.0.1 DNS 77 Standard query 0x2188 A cdn-0.nflximg.com
349 65.276870 192.168.0.1 192.168.0.21 DNS 489 Standard query response 0x2188 A cdn-0.nflximg.com CNAME images.netflix.com.edge...
350 65.277992 192.168.0.21 63.80.242.48 TCP 74 37063 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=491519482 TSecr=3295534150
351 65.297757 63.80.242.48 192.168.0.21 TCP 74 80 → 37063 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PERM=1 TSval=3295534150
352 65.298396 192.168.0.21 63.80.242.48 TCP 66 37063 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=491519502 TSecr=3295534130
353 65.298687 192.168.0.21 63.80.242.48 HTTP 153 GET /us/nrd/clients/flash/814540.bun HTTP/1.1
354 65.318730 63.80.242.48 192.168.0.21 TCP 66 80 → 37063 [ACK] Seq=1 Ack=88 Win=5792 Len=0 TSval=3295534151 TSecr=491519503
355 65.321733 63.80.242.48 192.168.0.21 TCP 1514 [TCP segment of a reassembled PDU]

> Frame 349: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits)
> Ethernet II, Src: Globalsec_00:3b:0a (f0:ad:4e:00:3b:0a), Dst: Vizio_14:8a:e1 (00:19:9d:14:8a:e1)
> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.21
> User Datagram Protocol, Src Port: 53 (53), Dst Port: 34036 (34036)
v Domain Name System (response)
  [Request In: 348]
  [Time: 0.034338000 seconds]
  Transaction ID: 0x2188
  > Flags: 0x8180 Standard query response, No error
  Questions: 1
  Answer RRs: 4
  Authority RRs: 9
  Additional RRs: 9
  v Queries
  > cdn-0.nflximg.com: type A, class IN
  > Answers
  > Authoritative nameservers
0020 00 15 00 35 04 f4 01 c7 83 3f 21 88 81 80 00 01 ...5....?!.....
0030 00 04 00 09 00 09 05 63 64 6e 2d 30 07 6e 66 6c .....c dn-0.nfl
0040 78 69 6d 67 03 63 6f 6d 00 00 01 00 01 c0 0c 00 ximg.com .....
0050 05 00 01 00 00 05 29 00 22 06 69 6d 61 67 65 73 .....). "images
0060 07 6e 65 74 66 6c 69 78 03 63 6f 6d 09 65 64 67 .netflix .com.edge
0070 65 73 75 69 74 65 03 6e 65 74 00 c 2f 00 05 00 esuite.n et....

Identification of transaction (dns.id), 2 bytes | Packets: 10299 * Displayed: 10299 (100.0%) * Load time: 0:0.182 | Profile: Default
```

Response Generation Module: Out of Domain Attacks

- Chatbot is generally very adept at a select few domains
 - What if we make out of domain requests?
- What are some potential solutions?
 - Train a classifier to detect out of domain requests
 - Improve network's ability to quantify uncertainty

Response Generation Module: Adversarial Text Samples

- Chatbot is constantly learning from its environment
 - What if we purposely poison the environment?
- What are some potential solutions?
 - Employ a hate speech detector

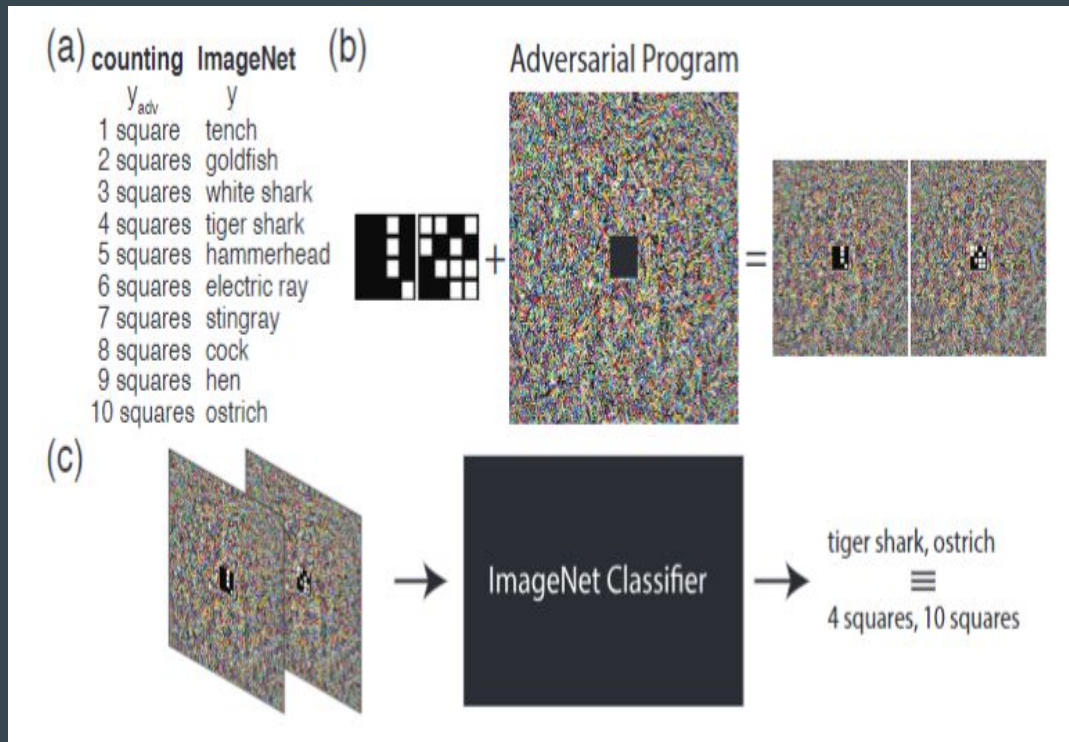


Response Generation Module: Language Model Attacks

- State of the art chatbots rely on language models like BERT
 - What if we can create malicious language models that sabotage the chatbot very discretely?
- What are some potential solutions?
 - Search for trigger words
 - Constantly vet language models

Response Generation Module: Adversarial Reprogramming

- The chatbot replies on a number of different deep learning modules
 - What if we can repurpose these modules for our malicious tasks?
- What are some potential solutions?
 - Make it harder for the adversary to learn the weaknesses of the model



Response Generation Module: Feedback Engineering

- Chatbot usually gets a reward signal from the user
- The system improves itself through either:
 - Retraining
 - Reinforcement learning
- What if we can discretely retrain the chatbot to use offensive language after hearing certain trigger words?
- What if we can alter the reward signal and get the chatbot to adopt our malicious policy?
- What are some possible solutions?
 - Make it harder to query the model multiple times
 - Separate training examples and response generation module

Database Module

- Database module houses a lot of sensitive information
 - What if we launch an injection attack against it?
 - What if we manipulate the knowledge graph?
- What are some possible solutions?
 - Search database for injection vulnerabilities before deployment
 - Clean the data used to train the knowledge graph

Conclusion

- Contact Information
 - wye@email.wm.edu