

# **ENEE 457: Computer Systems Security**

## **8/29/18**

### **Lecture 2**

### **Attacks from the Real World**

**Charalampos (Babis) Papamanthou**

Department of Electrical and Computer Engineering  
University of Maryland, College Park



# Attacks in the real world

WEB & COMMUNICATION SOFTWARE security  
**Hotmail Data Loss Reveals Cloud Trust Issues**  
Jan 3, 2011 11:56 AM  
By Keir Thomas, PCWorld

News  
**Amazon struggles to restore lost data to European cloud customers**  
Developers vent frustration on Amazon support forum  
By Jon Brodwin, Network World  
August 09, 2011 11:17 AM ET

## Gmail Corrupting Attachments

I recently received a report that attachments sent to Gmail from some servers

« [Security Recommendation](#) | [Main](#) | [Solaris Security](#)  
**Amazon S3 Silent Data Corruption**  
By user12606733 on Jan 28, 2009  
While catching up on my reading, I came across an [interesting article](#) focused on ti

01 August 2012, 12:39

## Dropbox confirms data leak

Cloud storage service provider [Dropbox](#) has [acknowledged](#) that a file

**BPOS: a data leak in Microsoft's cloud**  
December 28th, 2010 - 09:10 am ET by J. G.

A configuration error in Microsoft's Business Productivity

# Dropbox data loss

- Problem: Not sufficient back-ups
- How can you prevent this?
  - **ERROR CORRECTION**

One user, Michael Armogan, shared the contents of an email he received from Dropbox:

We're reaching out to let you know about a Selective Sync issue that affected a small number of Dropbox users. Unfortunately, some of your files were deleted when the Dropbox desktop application was shut down or restarted while you were applying Selective Sync settings.

Our team worked hard to restore files that were deleted from your account. You can see which of your files were affected and whether or not we've been able to restore them on this personalized web page.

We're very sorry about what happened. There's nothing more important to use than making sure your information is safe and always available. Our team has fixed the issue and put additional tests in place to prevent this from happening in the future.

# Error correcting codes

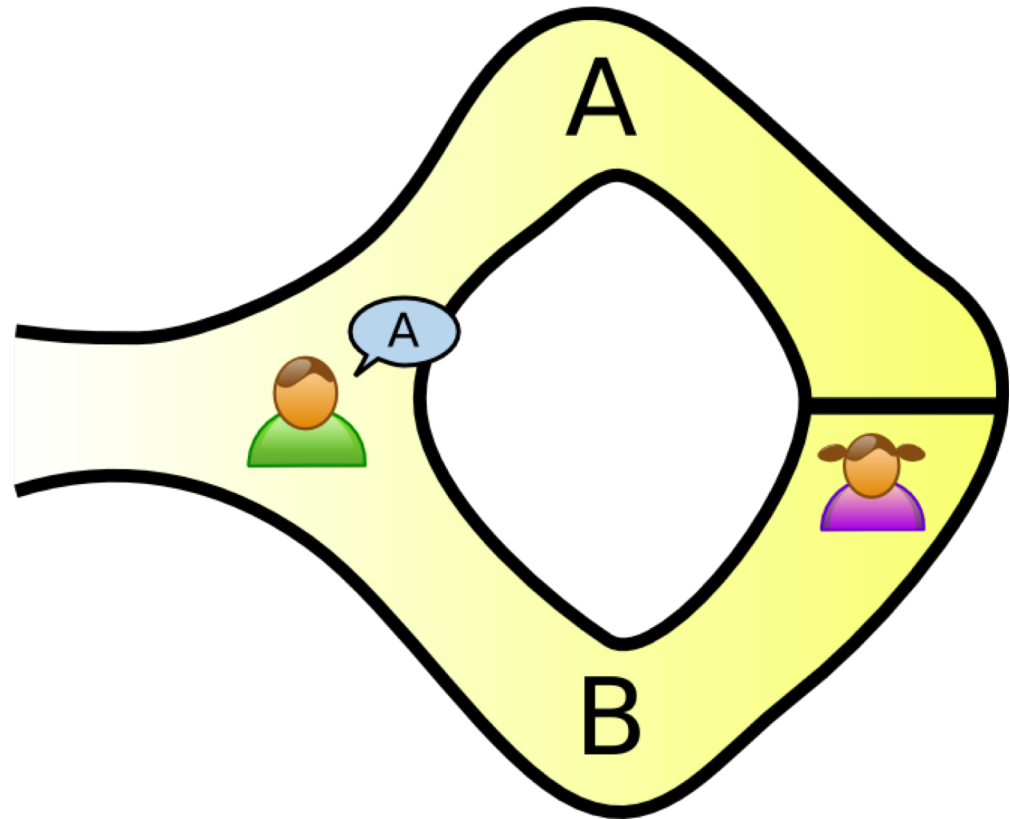
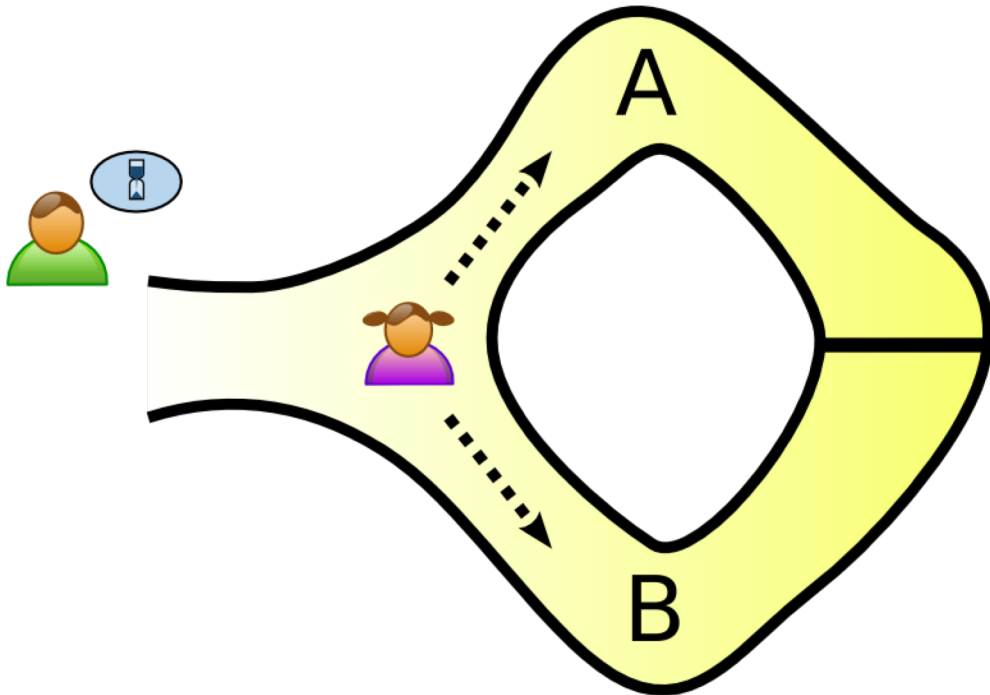
- Given blocks  $b_1, b_2, \dots, b_n$ , we want to store them with redundancy
- If we just duplicate them and store copies  $c_1, c_2, \dots, c_n$ , it is no good. The attacker can just delete 2 blocks (e.g.,  $b_1$  and  $c_1$ ) and cause damage
- Alternatively, consider the polynomial  $p(x) = (x - b_1)(x - b_2) \dots (x - b_n)$
- Store  $b_1, b_2, \dots, b_n$  and  $p(r_1), p(r_2), \dots, p(r_n)$  for random  $r_1, r_2, \dots, r_n$
- Now even if the attacker deletes **any**  $n - 1$  out of  $2n$  blocks that we have stored, we can always recover our initial data
- How? Polynomial interpolation!
- As long as  $n + 1$  out of  $2n$  points are stored intact, we can always recover the initial data

# LinkedIn passwords leaked

- In June 2012, it was announced that almost 6.5 million LinkedIn passwords were leaked and posted on a hacker site
  - [http://www.huffingtonpost.com/2012/06/07/linkedin-password-hack-check\\_n\\_1577184.html](http://www.huffingtonpost.com/2012/06/07/linkedin-password-hack-check_n_1577184.html)
  - Problem: LinkedIn did not use salt when hashing the passwords!
    - <http://www.stormpath.com/blog/how-linkedin-could-have-secured-hacked-passwords>
  - How can you prevent this?
    - **ALWAYS USE SALT**
- (when using salting, one cannot use preexisting tables to crack passwords easily)

# Teaser question about passwords

- Can you log into Google without sending your password over?
  - Zero-knowledge proofs!



# Speculative execution attacks on trusted hardware (from 10,000 feet)

- Researchers recently showed how to extract secret data from trusted hardware
- Paper (published in USENIX SECURITY 2018):  
[https://www.usenix.org/system/files/conference/usenixsecurity18/sec18-van\\_bulck.pdf](https://www.usenix.org/system/files/conference/usenixsecurity18/sec18-van_bulck.pdf)
- SGX supports “enclaves”, isolated computing environments that cannot be accessed by OS (e.g., they store secret keys)
- Application of SGX: Send encrypted data to a computer, have the computer do a computation, and return the result, without anyone seeing the data;
- Speculative execution feature used by Intel: If a then b. Execute a and b simultaneously and decide in the end whether you will do a roll-back or not; Mega-performance boost
- Attacker: Read Enclave Memory
- SGX: If attacker is allowed, then return SGX memory. Intel runs this speculatively
- So attacker gets to see SGX memory before SGX protection mechanisms kick in and undo the result.

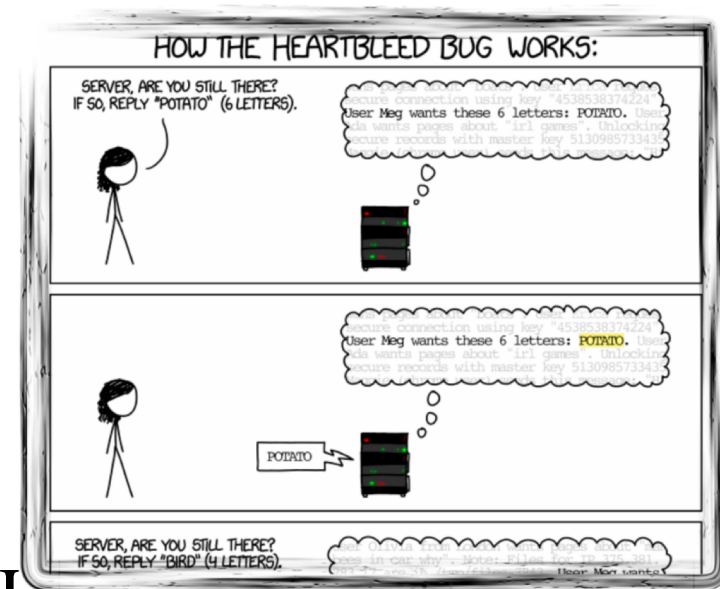
# Factoring RSA keys

- Researchers recently showed that a bunch of cryptographic keys used in hardware devices are insecure
- Companies shipped new updates after notified
- <https://factorable.net/>
- Problem: Same randomness used across devices to generate the keys
- How can you prevent this?
  - **MAKE SURE YOU USE DIFFERENT SEEDS FOR YOUR RANDOM GENERATORS**



# Heartbleed

- April 2014
- Bug in the openssl library
- Affected all hosts running TLS protocol
- At the time of the disclosure, around half a million of the Internet's secure web servers certified were believed to be vulnerable to the attack
- Bug in the heartbeat feature <http://tools.ietf.org/pdf/rfc6520.pdf>
- There was no bound check in the bytearray that the sender would send to the receiver
- So the receiver would send the payload back along with some contents of its memory
- How can you prevent this?
  - **CHECK ARRAY BOUNDS**



# WEP standard CRC check

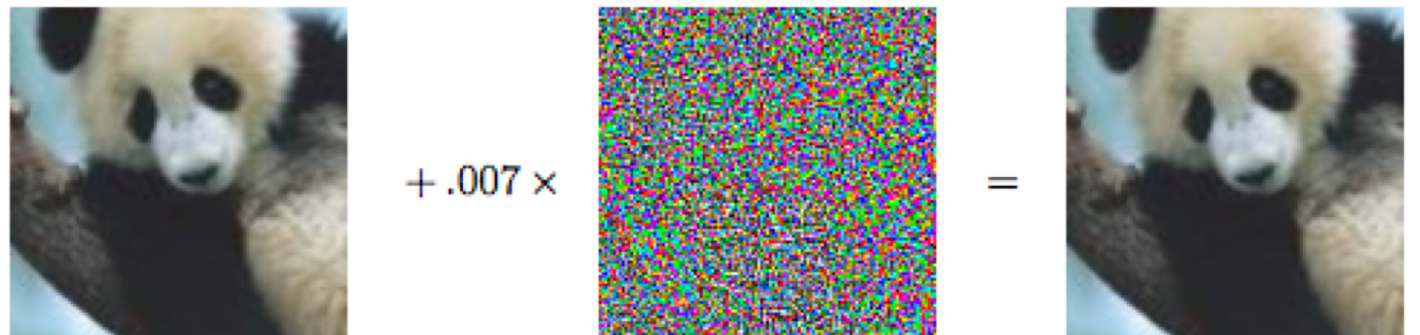
- Security algorithm for privacy and integrity of wi-fi communications
- Introduced in 1997
- Completely broken
- To send a message 1011101:
  - Represent it as a polynomial ( $x^6+x^4+x^2+x+1$ )
  - Divide with  $x^2+1$
  - Send the remainder (010) and the message
  - The receiver takes the message and can verify
- What is the problem here?
- How can you prevent this?
  - **USE A CRYPTOGRAPHICALLY SECURE MESSAGE AUTHENTICATION CODE**

# Order Preserving Encryption

- Type of deterministic encryption such that
  - If  $x > y$  then  $\text{Enc}(x) > \text{Enc}(y)$
  - Very useful for encrypted databases
  - Used by many systems such as CryptDB, Cipherbase, Google's BigQuery and Microsoft SQL Always Encrypted
- First problem: Deterministic encryption
  - Why this is bad?
  - If I give you bunch of ciphertexts that are names, how can you decrypt?
- Second problem: The order is revealed
  - How can you do better?
- See recent paper describing attack paper <http://cs.brown.edu/~seny/pubs/edb.pdf>
- How can you prevent this?
  - **USE RANDOMIZED ENCRYPTION...BUT?**

# Adversarial Machine Learning

- Confusing Neural Networks



$x$   
“panda”  
57.7% confidence

$+ .007 \times$

$\text{sign}(\nabla_x J(\theta, x, y))$   
“nematode”  
8.2% confidence

$=$

$x + \epsilon \text{sign}(\nabla_x J(\theta, x, y))$   
“gibbon”  
99.3 % confidence

- But for self driving cars: What if I place an adversarial sign at some intersection?

# Targeted vs Untargeted Attack

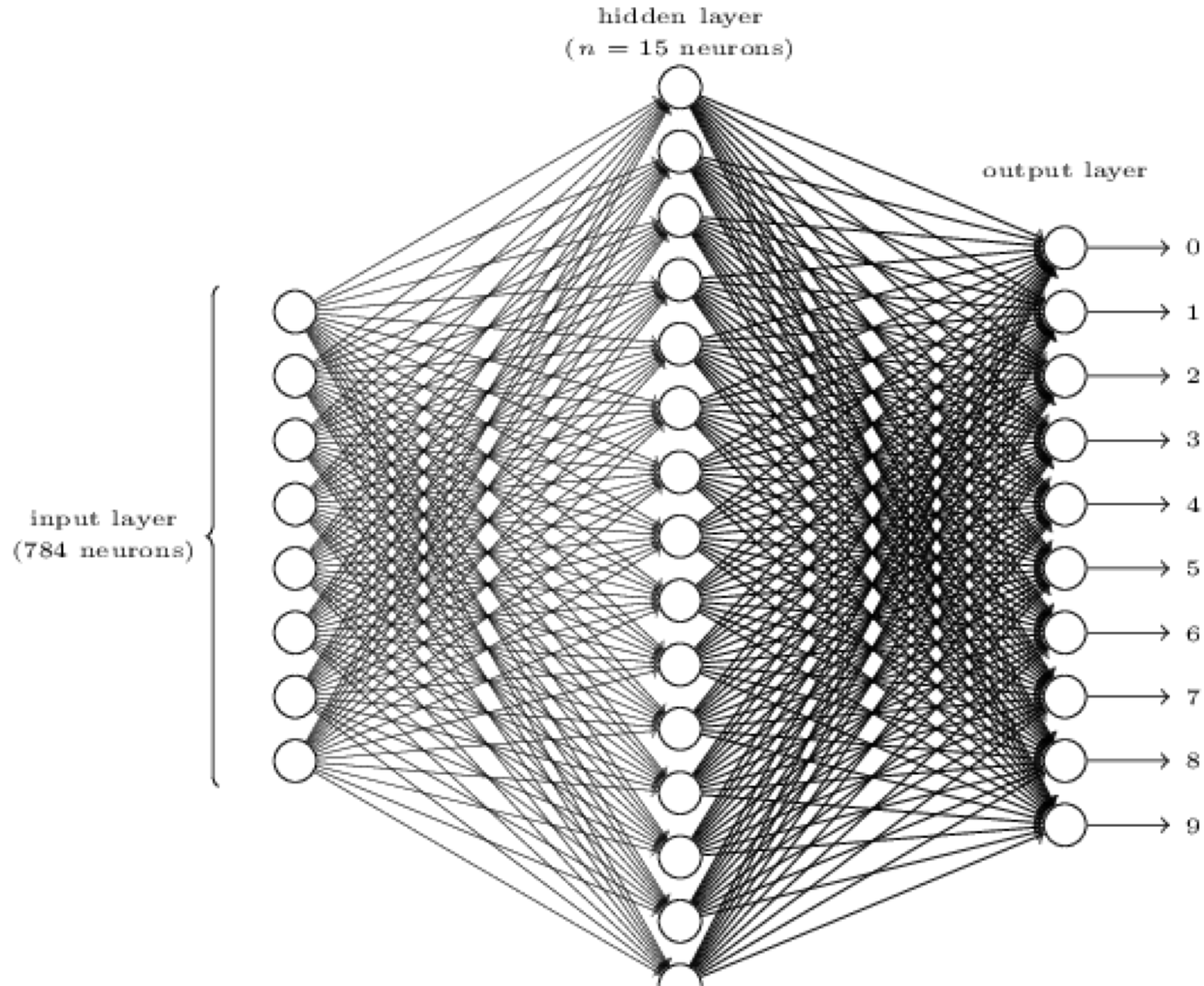
- Stop sign example is a targeted attack: I must create a sign that looks like stop otherwise people will catch me fast
- Face ID feature in Iphone X uses a neural network. There I am looking to create any image that can bypass the neural network. This is an non-targeted attack.

# Neural Network for recognizing hand-written digits

- Training: Provide pairs of handwritten digits and correct labels and figure out the function
- Testing. Provide an arbitrary hand-written digit and let the network figure it out
- Every digit is 28 x 28 pixels. Each pixel is 256 bits

A row of six handwritten digits: 5, 0, 4, 1, 9, and 2. The digits are drawn in a simple, slightly irregular black ink on a white background, typical of the MNIST dataset.

# How to classify handwritten digits



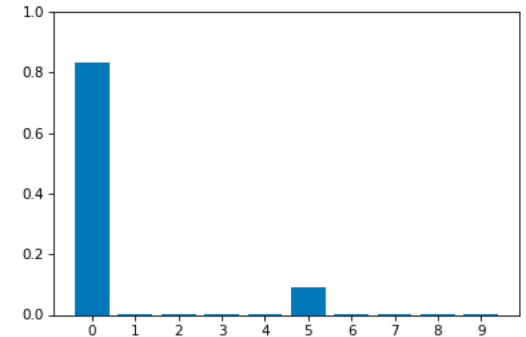
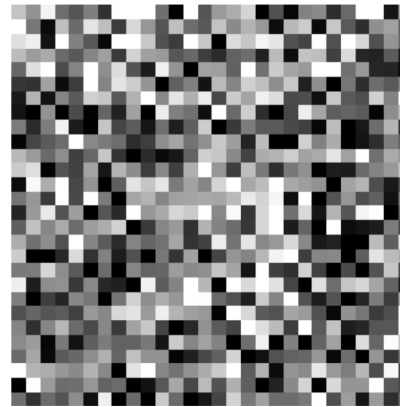
- Every neuron has bias  $b$
- Every edge has weight  $w_i$
- Inputs are 784 neurons in  $[0,1]$
- Outputs are 10 labels in  $[0,1]$
- The label that the network thinks is correct is the one with the biggest  $p$  value
- Use
$$f(x) = \frac{1}{1 + e^{-x}}$$
- Where  $x = \text{SUM}(x_i w_i) + b$

# Launch a non-targeted attack

- Goal vector
- Function to be minimized!
- Output

$$y_{goal} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$C = \frac{1}{2} \|y_{goal} - \hat{y}(\vec{x})\|_2^2$$



- Why this is an attack?
- What if the  $x$  that minimizes is actually close to the digit 0?
- Note that similar technique is used for training. How?



# How to minimize?

- One can take derivatives
- But solving these equations  $= 0$  is complicated due to the sigmoid function
- Instead we can use a method called gradient descent

Gradient descent is based on the observation that if the multi-variable function  $F(\mathbf{x})$  is defined and differentiable in a neighborhood of a point  $\mathbf{a}$ , then  $F(\mathbf{x})$  decreases *fastest* if one goes from  $\mathbf{a}$  in the direction of the negative gradient of  $F$  at  $\mathbf{a}$ ,  $-\nabla F(\mathbf{a})$ . It follows that, if

$$\mathbf{a}_{n+1} = \mathbf{a}_n - \gamma \nabla F(\mathbf{a}_n)$$

for  $\gamma$  small enough, then  $F(\mathbf{a}_n) \geq F(\mathbf{a}_{n+1})$ . In other words, the term  $\gamma \nabla F(\mathbf{a})$  is subtracted from  $\mathbf{a}$  because we want to move against the gradient, toward the minimum. With this observation in mind, one starts with a guess  $\mathbf{x}_0$  for a local minimum of  $F$ , and considers the sequence  $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \dots$  such that

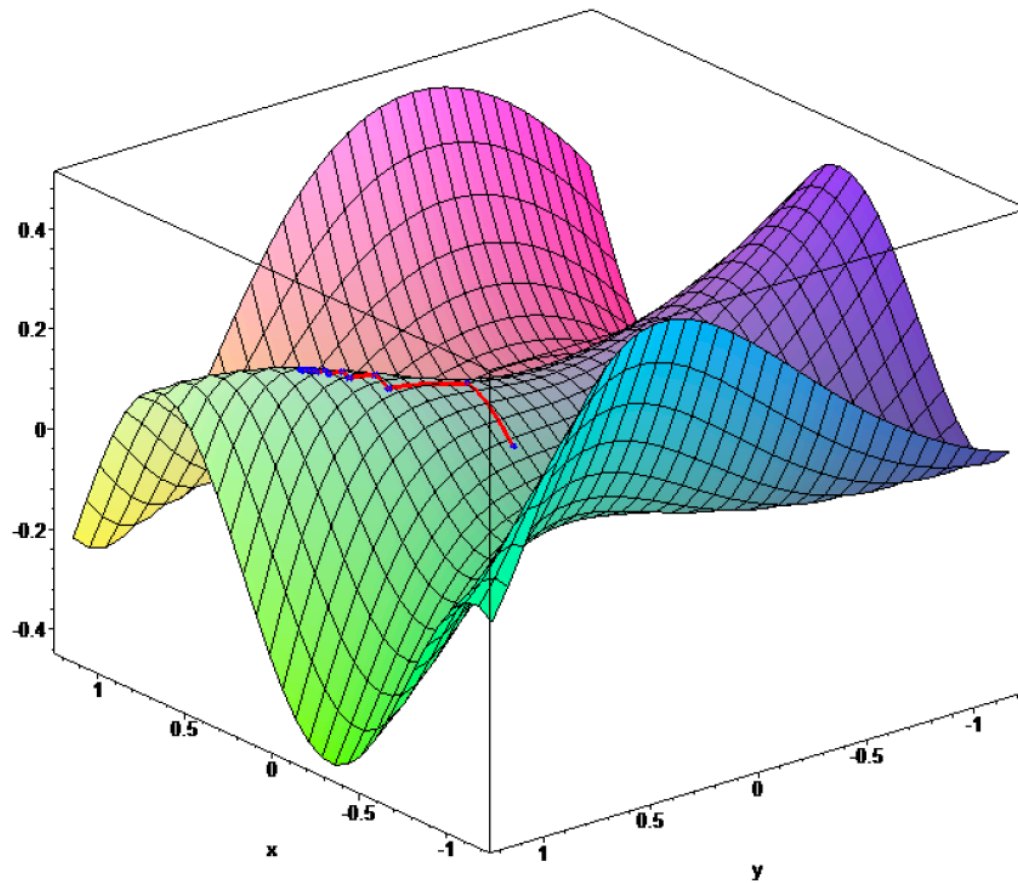
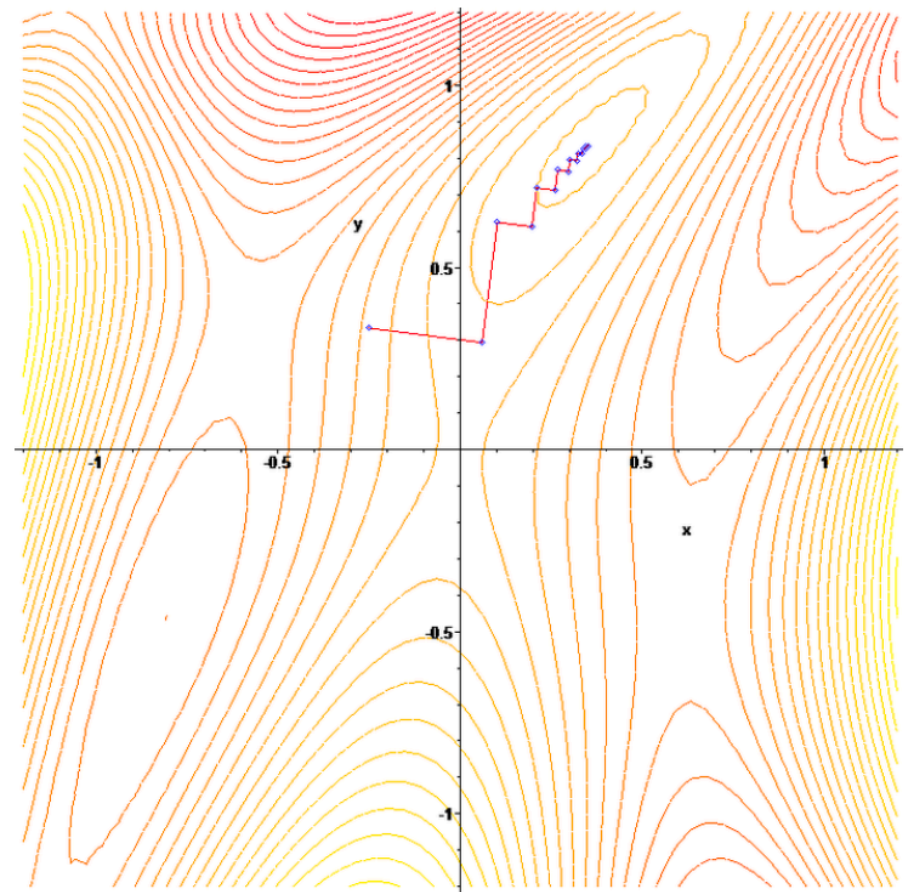
$$\mathbf{x}_{n+1} = \mathbf{x}_n - \gamma_n \nabla F(\mathbf{x}_n), \quad n \geq 0.$$

We have

$$F(\mathbf{x}_0) \geq F(\mathbf{x}_1) \geq F(\mathbf{x}_2) \geq \dots,$$

# Example of gradient descent

The "Zig-Zagging" nature of the method is also evident below, where the gradient descent method is applied to  $F(x, y) = \sin\left(\frac{1}{2}x^2 - \frac{1}{4}y^2 + 3\right) \cos(2x + 1 - e^y)$ .



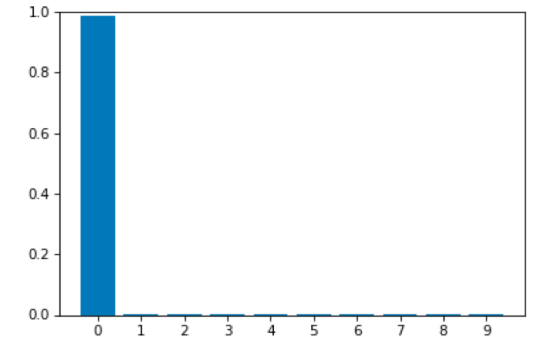
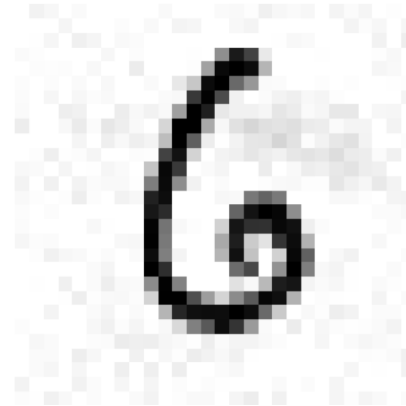
# Launch targeted attack

- Goal vector
- Function to be minimized! ( $x_{\text{target}}$  is handwritten 0 (different that 5))

$$y_{\text{goal}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

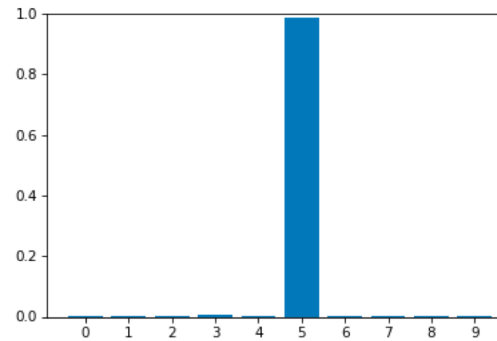
$$C = \frac{1}{2} \|y_{\text{goal}} - \hat{y}(\vec{x})\|_2^2 + \lambda \|\vec{x} - x_{\text{target}}\|_2^2$$

- Output



# Protection

- Note that most pics produced by targeted attack have some gray in the background



- Remove gray, but problem might be:



# Black box attacks

- Produce attacks for one neural, work for others!
- See paper <https://arxiv.org/pdf/1611.02770.pdf>

# **Some Security Principles to minimize exposure to attacks**

# Economy of mechanism

- This principle stresses simplicity in the design and implementation of security measures
  - Example: Avoid multiple interconnecting software modules (running in different machines) to implement a security property (e.g., input of password in one machine, checking of password in another)

# Complete mediation

- The idea behind this principle is that every access to a resource must be checked for compliance with a protection scheme
  - As a consequence, one should be wary of performance improvement techniques that save the results of previous authorization checks, since permissions can change over time
  - For example, an online banking web site should require users to sign on again after a certain amount of time, say, 15 minutes, has elapsed



# Open design

- According to this principle, the security architecture and design of a system should be made publicly available
  - Security should rely only on keeping cryptographic keys secret
  - Open design allows for a system to be scrutinized by multiple parties, which leads to the early discovery and correction of security vulnerabilities caused by design errors
  - The open design principle is the opposite of the approach known as security by obscurity, which tries to achieve security by keeping cryptographic algorithms secret and which has been historically used without success by several organizations

# Fail-safe defaults

- If a system fails, it should return to the most secure mode
- Example: How do firewalls work?
  - They receive packets
  - They make decision whether to forward or drop the packet
  - If a firewall is attacked, or fails, it should ALWAYS reject a packet
  - This is good for security (at that point, functionality is gone anyways!)

# Separation of responsibility

- Split a privilege so that at least two parties need to agree for an action to be performed
  - In a theatre you have two people making sure you are allowed to get in
    - A cashier that sells you the ticket and issues a receipt
    - Another employee outside of the performance room that checks the ticket
- Why this is good?
  - Cashiers are low-paid employees that might be tempted to do a favor to a friend
  - The other employee keeps them honest
- Computer system analogy
  - Check for access control in two places, so that if one gets compromised (e.g., through a buffer overflow), the other will still work

# Design security in from the start

- Internet is a bad example of that
  - Now we have to patch it up (DNSSEC, IPSEC)
- Do not focus only on functionality, especially when you design network applications
  - Assume the attacker will get you
  - If something can go wrong, it will...

# Psychological acceptability

- This principle states that user interfaces should be well designed and intuitive, and all security-related settings should adhere to what an ordinary user might expect
  - E.g., if a security administrator asks you to pick a 17 letters completely random password and change it every month, most likely you will write it down on a piece of paper
  - If you take 3 minutes to figure out how to set up secure connection with your friend, probably you will communicate insecurely

# Work factor

- According to this principle, the cost of circumventing a security mechanism should be compared with the resources of an attacker when designing a security scheme
  - A system developed to protect student grades in a university database, which may be attacked by snoopers or students trying to change their grades, probably needs less sophisticated security measures than a system built to protect military secrets, which may be attacked by government intelligence organizations

# Compromise recording (better to detect than to prevent)

- This principle states that sometimes it is more desirable to record the details of an intrusion than to adopt more sophisticated measures to prevent it
  - Internet-connected surveillance cameras are a typical example of an effective compromise record system that can be deployed to protect a building in lieu of reinforcing doors and windows
  - The servers in an office network may maintain logs for all accesses to files, all emails sent and received, and all web browsing sessions