# 1st Meeting Schedule

<table>
<thead>
<tr>
<th></th>
<th>Mon</th>
<th>Tue</th>
<th>Wed</th>
</tr>
</thead>
<tbody>
<tr>
<td>21-Oct</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3:00pm - 3:30pm</td>
<td>Sungchul Lee</td>
<td>Robert Gripentog</td>
<td>Candace Suh</td>
</tr>
<tr>
<td>3:30pm - 4:00pm</td>
<td></td>
<td>Sairam Ganti</td>
<td></td>
</tr>
<tr>
<td>4:00pm - 4:30pm</td>
<td>Govind Pathak</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4:30pm - 5:00pm</td>
<td>Siavash Akrami</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27-Oct</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3:00pm - 3:30pm</td>
<td>Djona Fegnem Eric</td>
<td>Sweta Gurung</td>
<td>Tara Baniya</td>
</tr>
<tr>
<td>3:30pm - 4:00pm</td>
<td>Alexis Fouche</td>
<td>Anusha Ramanathan</td>
<td>Vedaradatta Gouripeddi</td>
</tr>
<tr>
<td>4:00pm - 4:30pm</td>
<td>Marc Dinh</td>
<td>Monetta Shaw</td>
<td>Vikek Gudibande</td>
</tr>
<tr>
<td>4:30pm - 5:00pm</td>
<td></td>
<td>Kalpana Rajagopal</td>
<td>Wilfred Cabalo</td>
</tr>
</tbody>
</table>

Meetings are in my office, SEB 4245
(Need to check in 1st floor)
# 2nd Meeting Schedule

<table>
<thead>
<tr>
<th></th>
<th>Mon</th>
<th>Wed</th>
<th>Thu</th>
</tr>
</thead>
<tbody>
<tr>
<td>10-Nov</td>
<td>3:00pm</td>
<td>Govind Pathak</td>
<td>Robert Gripentog</td>
</tr>
<tr>
<td></td>
<td>3:30pm</td>
<td>Robert Gripentog</td>
<td>Govind Pathak</td>
</tr>
<tr>
<td>12-Nov</td>
<td>3:30pm</td>
<td>Sairam Ganti</td>
<td>Tara Baniya</td>
</tr>
<tr>
<td>13-Nov</td>
<td>4:00pm</td>
<td>Tara Baniya</td>
<td>Anusha Ramanathan</td>
</tr>
<tr>
<td></td>
<td>4:30pm</td>
<td>Anusha Ramanathan</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Mon</th>
<th>Wed</th>
<th>Thu</th>
</tr>
</thead>
<tbody>
<tr>
<td>17-Nov</td>
<td>3:00pm</td>
<td>Sungchul Lee</td>
<td>Marc Dinh</td>
</tr>
<tr>
<td></td>
<td>3:30pm</td>
<td>Sungchul Lee</td>
<td>Marc Dinh</td>
</tr>
<tr>
<td>19-Nov</td>
<td>3:30pm</td>
<td>Veddatta Gourippedi</td>
<td>Kalpana Rajagopal</td>
</tr>
<tr>
<td></td>
<td>4:00pm</td>
<td>Veddatta Gourippedi</td>
<td>Kalpana Rajagopal</td>
</tr>
<tr>
<td>20-Nov</td>
<td>4:00pm</td>
<td>Winfred Cabalo</td>
<td>Monetta Shaw</td>
</tr>
<tr>
<td></td>
<td>4:30pm</td>
<td>Winfred Cabalo</td>
<td>Monetta Shaw</td>
</tr>
<tr>
<td></td>
<td>4:30pm</td>
<td>Alexis Fouche</td>
<td>Vivek Gudibande</td>
</tr>
</tbody>
</table>
Lab: Wireless Security with Kali

- Wireshark
Lab: Wireless Security with Kali

❖ Kismet

![Kismet Sort View Windows](image)
Networking

- View and change configuration
  
  ifconfig
  
  (lo for local loopback interface, eth0 for ethernet)

  iwconfig

- View status
  
  netstat -na or

  lsof -i (list open files)

  lsof -p pid (shows all the files the process if using)
Wireless Security Protocols

Failure in WEP
WEP Encryption Process

The WEP encryption process involves the following steps:

1. **PRNG (Pseudo-Random Number Generator)**: Generates a key stream.
2. **IV (Initial Vector)**: A unique number used to start the encryption process.
3. **Data + ICV (Integrity Check Value)**: Combined with the key stream using the XOR operation.
4. **802.11 Frame Payload**: The encrypted data is then inserted into the 802.11 frame payload.
5. **802.11 Header** and **802.11 Trailer**: These encapsulate the encrypted data for transmission.

The process ensures that only authorized parties with the correct IV and WEP key can decrypt the data, providing a basic level of security for wireless networks.
WEP Decryption Process

![Diagram of WEP Decryption Process]

- **802.11 header**
- **IV**
- **Other**
- **Data**
- **ICV**
- **802.11 trailer**

- **IV+WEP key**
- **PRNG**
- **Key stream (XOR)**
- **Encrypted[Data+ICV]**
- **Data**
- **ICV**
Why did WEP fail?

1. No replay protection (can use same IV)
2. Weak message integrity check (used CRC-32)  
   ✗ bit flipping possible
3. No key rotation mechanism
4. Initialization vector is too short (24 bits)
5. Challenge/response reveals PRGA
6. Key is reversible from cipher text
WEP Failure 4: Initialization Vector Too Short

- RC4 requires each key to be unique → IV is needed
  - IV transmitted in clear!
- Only 16,777,216 possible IV’s ($2^{24}$)
  - For multiple hosts, not a large number
  - Theoretical packet rate in 802.11b: about 6,000 packets/sec
- Chances of collision are very strong
  - Birthday attack, 23 people, 50/50
  - 50% chance after 4,823 packets
  - 99% chance after 12,430 packets
  - Even with 128-bit key, the IV is still 24 bits – No better!
  - Sequential IV selection, easy attack
    - multiple machines booting around same time may use the same range of IV sequentially
    - AP reboots create lot of collision
Detecting IV Collisions

- Kismet reports these for each BSSID observed (network details – “i” key)
  - Packets
    - Data
    - LLC
    - Crypt
    - Weak
    - Dupe IV
IV Collision – Keystream Attack

☐ Recall:
  ☐ KSA generates per-packet WEP key from (PSK + IV)
  ☐ When an **IV is used twice**, the same KSA is used -> not an one-time pad any more!
  ☐ If there is a **duplicate IV**, it is possible for the attacker to find out plaintext of one of the encrypted packets if the plaintext of the colliding ciphertext is known

☐ IV collision reveals keystream with known plaintext

☐ Two cipher – texts and one plaintext known, plaintext can be derived

☐ Partial known plaintext is sufficient for attack

\[ \text{Cipher}_1 \oplus \text{Cipher}_2 = \text{Plain}_1 \oplus \text{Plain}_2 \]
## Review - Using the same key?

If \((a \oplus k) = A\), and \((b \oplus k) = B\),

then \(A \oplus B = (a \oplus k) \oplus (b \oplus k)\)

\[= a \oplus b \oplus k \oplus k\]

\[= a \oplus b \oplus 0\]

<table>
<thead>
<tr>
<th>(A)</th>
<th>(B)</th>
<th>XOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

---

**e.g., Verona Project**
## IV Collision – Recovering Plaintext

<table>
<thead>
<tr>
<th>Plaintext 1</th>
<th>PRGA 1</th>
<th>Ciphertext 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 1 1 1 0 1 1 0</td>
<td>1 1 0 0 0 1 1 0</td>
<td>1 0 1 1 0 0 0 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Plaintext 2</th>
<th>PRGA 1</th>
<th>Ciphertext 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0 0 0 0 1 1 1</td>
<td>1 1 0 0 0 1 1 0</td>
<td>0 1 0 0 0 0 0 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ciphertext 1</th>
<th>1 0 1 1 0 0 0 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ciphertext 2</td>
<td>0 1 0 0 0 0 0 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ciphertext 1</th>
<th>1 1 1 1 0 0 0 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ciphertext 2</td>
<td>1 1 1 1 0 0 0 1</td>
</tr>
</tbody>
</table>

**Attacker knows ciphertext 1 and 2, can calculate XOR product.**
If plaintext 1 is known, plaintext 2 can be determined.
Practical IV Collision Attack

- Windows DHCP request packets
  - Sent to broadcast address
  - Consistently 368 bytes, (WEP + 802.11)
  - Frame format produces consistent plaintext

- Attacker records IV for this packet
  - Waits for collision IV
  - Can decrypt 2\textsuperscript{nd} packet contents

Wireshark Filter: “frame.pkt_len eq 362 and wlan.da eq ff:ff:ff:ff:ff”
WEP Failure 5: Challenge/Response Reveals PRGA

- WEP authentication review:
  - AP issues 128-byte challenge to STA (cleartext nonce)
  - STA encrypts challenge, returns to AP (encrypted with WEP key)

- Attacker can monitor this exchange
  - Determines plaintext (challenge) and ciphertext (response)

- It does not reveal the PSK, but allows arbitrary packet injection

```
Challenge ⊕ Response = PRGA
```
WEP Authentication

Diagram:
- Station
  - Authentication Request
  - Challenge Text
  - Challenge Response (Encrypted Challenge Text)
  - Confirm Success
- Access Point
Attacking Challenge/Response

- Attacker determines 128-bytes PRGA
- Cannot decrypt traffic contents
- Crafts custom packets to broadcast
  - Generates expectable response (ping size)

- Uses PRGA to encrypt packet
- Uses matching IV to inject packet
  - E.g., ICMP echo (ping) with unusual packet size
  - Useful to increase the level of activity on the wireless network
    → increase likelihood of duplicate and weak IV

- Watches for responses
WEP Failure 6: Key is Reversible From Cipher Text

- Weak IV selection leads to key recovery
  - Despite the recommendation of RC4, WEP uses weak IVs
- Fluhrer, Mantin and Shamir (FMS) attack
  - Cryptographically weak IV selection
- Known plaintext reveals key information
  - 802.11 protocol uses 802.2 header immediately following the 802.11 header to identify payload type
  - First two bytes of WEP payload are constant 802.2 header (0xAA 0xAA)
- Enough weak IV’s recovers key
  - One weak IV can reveal a correct key byte 5% of the time.
  - With a large number of IVs the most probable key can be guessed
What are Weak IVs?

  - “B” represents the byte of the WEP key we are trying to recover, plus 3.
  - “FF” is consistent
  - N is any value

- **Some weak IV examples:**
  - 0x03 0xff 0xa5 = 262053
  - 0x06 0xff 0x1b = 458523
  - 0x04 0xff 0x7e = 327550
Tews, Weinmann, and Pyshkin

- “Breaking 104 bit WEP in less than 60 seconds” (2007)

- Discusses how to discover the RC4 key by analyzing the easily identified ARP packets.

- Aircrack-ng uses ptw method (ptw.cap)
Attacks Against WEP

- Vendor implementation weaknesses
- Dictionary key attacks
- FMS attack implementations
- Traffic replay attacks
Dictionary Attacks Against WEP
Attacking Key Selection

☒ Some vendors restrict key selection to ASCII characters
  ☒ Some people choose dictionary words
  ☒ Mount dictionary attack against key

☒ Uses IV and word to compute PRGA
  ☒ Resource intensive (CPU, disk)
  ☒ Only requires 2 packets (2nd packet, just to confirm)
FMS Attacks - AirSnort

- Early FMS attack implementation
- Recovers key after hundreds of thousands of packets
- Tries to recover key while capturing
  - Stops capture when key is recovered
FMS Attacks - Aircrack

- Recognizes 5.5 million weak IV’s
  - Effective against IV-filtered networks

- Requires ~250,000 packets to recover key
  - If lucky ~75,000
## Aircrack Example

- 5 byte key

```bash
$ aircrack -n 5 aircrack-data.dump

Choosing first encrypted BSSID = 00:0D:29:4A:B8:5A (KeyID 0)
Reading packets: total = 152633, usable = 75813.

aircrack 1.4

* Got 75813! Unique IVs | fudge factor = 2
* Elapsed time [00:00:01] | tried 37 keys at 2220 k/m

<table>
<thead>
<tr>
<th>KB</th>
<th>depth</th>
<th>votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1/ 5</td>
<td>88( 15) C3( 15) F4( 15) C7( 12) 43( 5) E9( 5)</td>
</tr>
<tr>
<td>1</td>
<td>0/ 1</td>
<td>85( 28) 6B( 13) 89( 13) 8F( 12) 1A( 5) 2E( 5)</td>
</tr>
<tr>
<td>2</td>
<td>1/ 3</td>
<td>74( 15) AC( 12) 87( 5) 94( 5) AE( 5) FF( 5)</td>
</tr>
<tr>
<td>3</td>
<td>0/ 2</td>
<td>27( 23) 01( 15) 22( 6) 37( 5) 38( 5) B7( 5)</td>
</tr>
<tr>
<td>4</td>
<td>4/ 7</td>
<td>35( 10) 8A( 9) E2( 8) 13( 5) 2A( 5) C3( 5)</td>
</tr>
</tbody>
</table>

KEY FOUND! [88:85:74:27:35]
```
How many APs use encryption?

- [ ] http://wigle.net/gps/gps/main/stats/
Lab: Wireless Security with Kali

- Aircrack
- Fern-wifi-cracker
- Wifite