Cracking WEP Keys

Applying known techniques to WEP Keys

Tim Newsham

@stake

Where Security & Business Intersect[SM]

Introduction

- **Developed WEP key cracking software**
  - Dictionary attack on the key generators
  - Dictionary attack on raw keys
  - Brute force of the 64-bit key generator
- **Analyzed Key Generators**
- **Did not perform new cryptanalysis on the WEP protocol**
- **Did not look at 802.1x and Radius**

Talk overview

- Motivation
- WEP protocol overview
- WEP keying
- WEP key generators
- A WEP Cracker
- Results
- Related Work

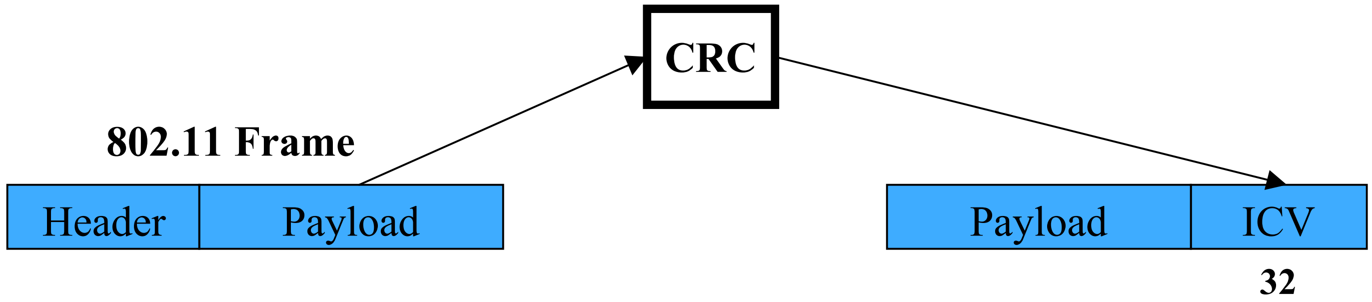
Why Perform Dictionary attacks on WEP?

- **Security is as good as the weakest link**
- **Key cracking attacks the human problem**
- **But Isn't WEP already broken?**
  - Key cracking is often simpler to implement and perform
  - Key cracking can be less time consuming

Wired Equivalent Privacy

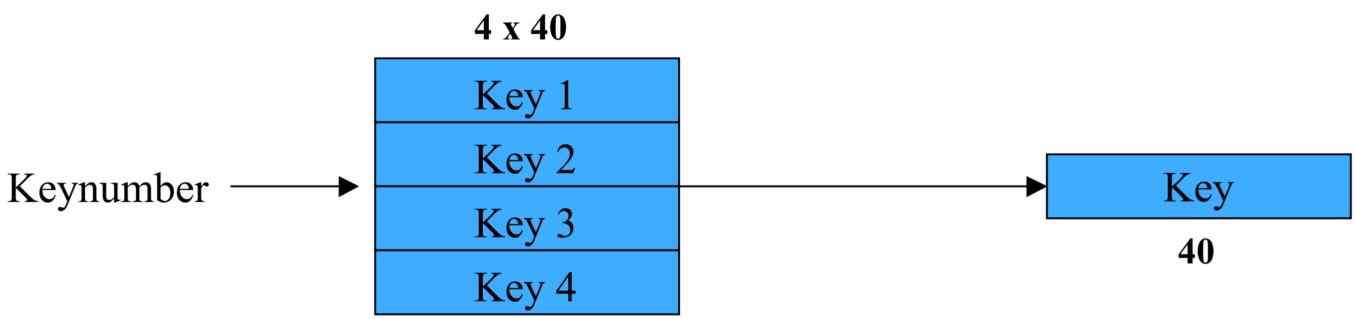
- **Purpose – bring the security of wired networks to 802.11**
- **Provides Authentication and Encryption**
- **Uses RC4 for encryption**
  - 64-bit RC4 keys
  - Non-standard extension uses 128-bit keys
- **Authentication built using encryption primitive – Challenge/Response**

WEP Encryption



- **ICV computed – 32-bit CRC of payload**

WEP Encryption



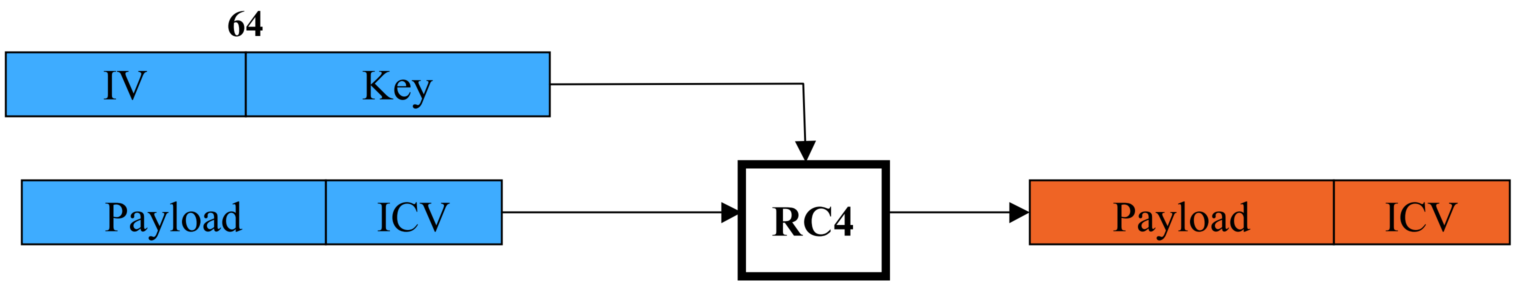
- ICV computed – 32-bit CRC of payload
- One of four keys selected – 40-bits

WEP Encryption

IV	keynumber
24	**8**

- **ICV computed – 32-bit CRC of payload**
- **One of four keys selected – 40-bits**
- **IV selected – 24-bits, prepended to keynumber**

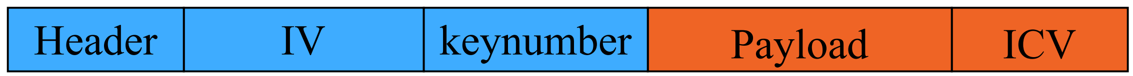
WEP Encryption



- ICV computed – 32-bit CRC of payload
- One of four keys selected – 40-bits
- IV selected – 24-bits, prepended to keynumber
- IV+key used to encrypt payload+ICV

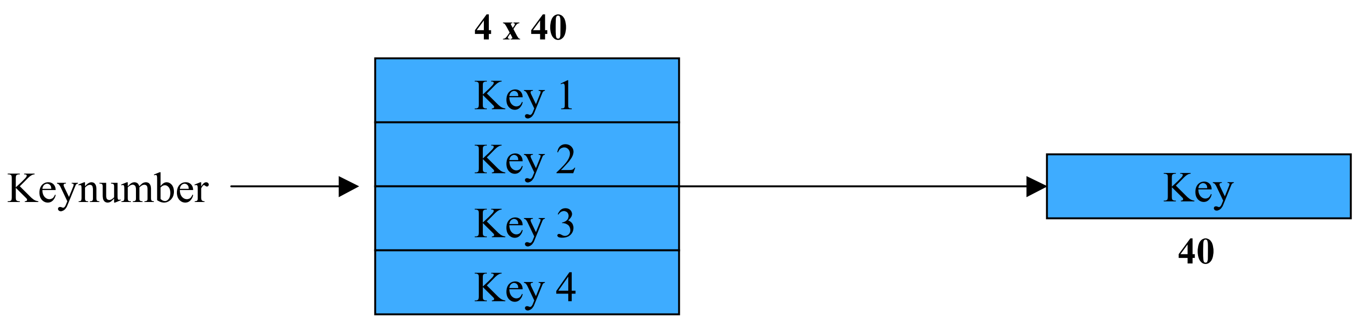
WEP Encryption

WEP Frame



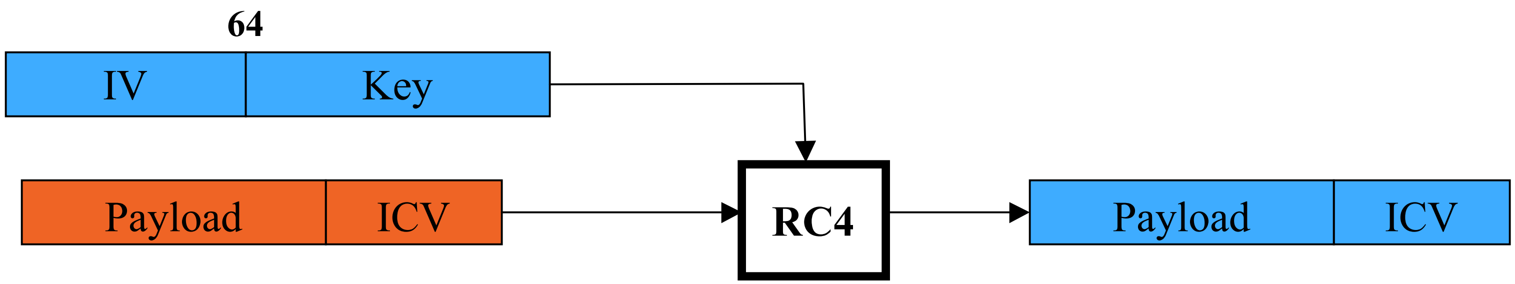
- ICV computed – 32-bit CRC of payload
- One of four keys selected – 40-bits
- IV selected – 24-bits, prepended to keynumber
- IV+key used to encrypt payload+ICV
- IV+keynumber prepended to encrypted payload+ICV

WEP Decryption



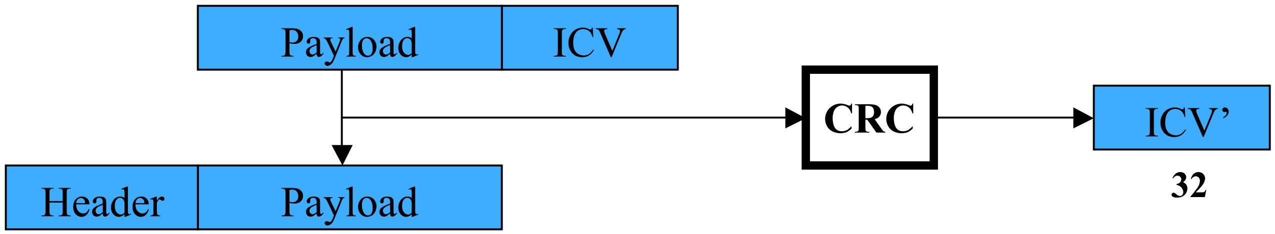
- **Keynumber is used to select key**

WEP Decryption



- Keynumber is used to select key
- ICV+key used to decrypt payload+ICV

WEP Decryption

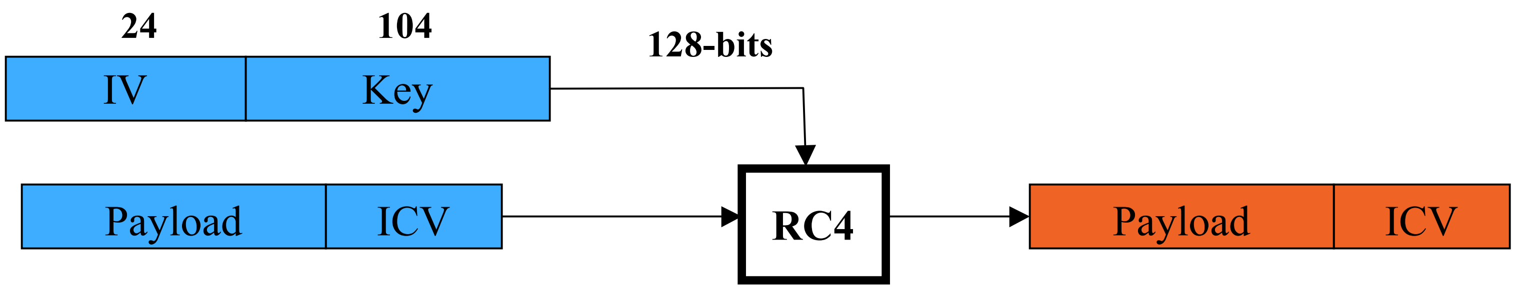


- Keynumber is used to select key
- ICV+key used to decrypt payload+ICV
- ICV recomputed and compared against original

WEP Authentication

- **Uses WEP encryption primitives**
  - Nonce is generated and sent to client
  - Client encrypts nonce and sends it back
  - Server decrypts response and verifies that it is the same nonce.
- **Authentication is optional**

128-bit Variant

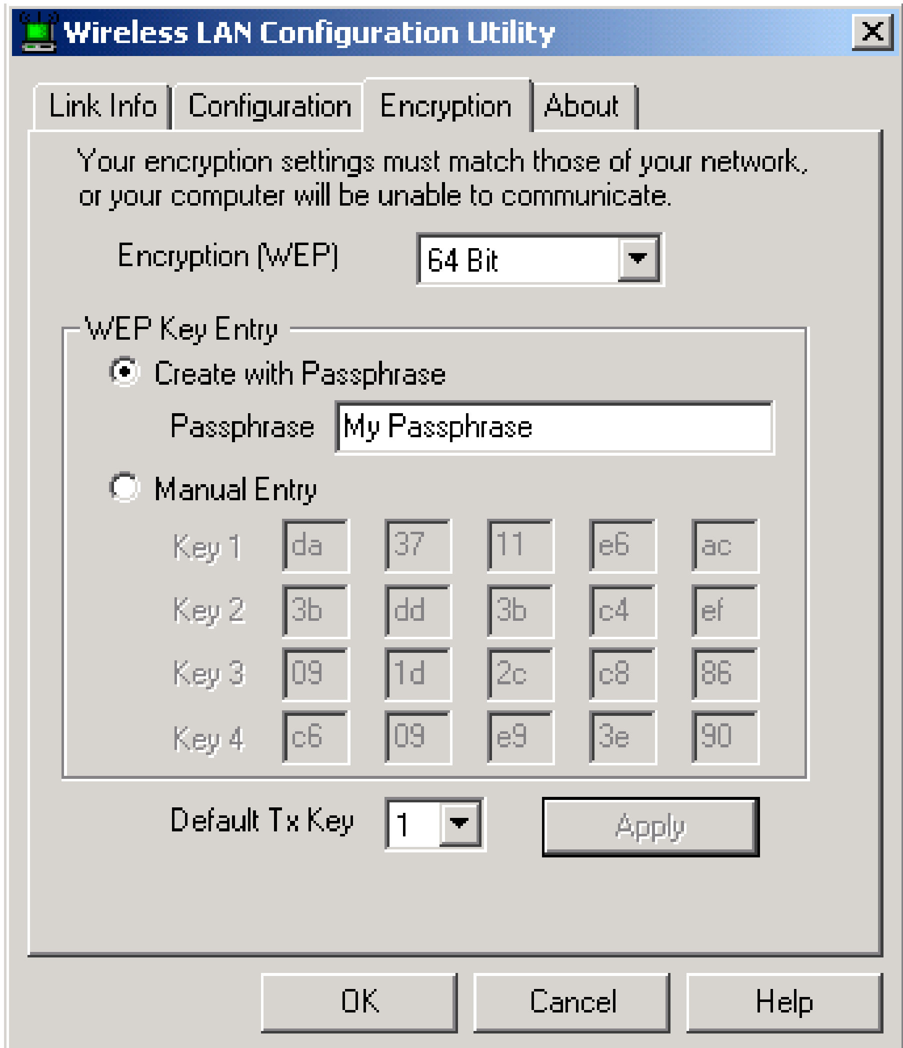


- Purpose – increase the encryption key size
- Non-standard, but in wide use
- IV and ICV set as before
- 104-bit key selected
- IV+key concatenated to form 128-bit RC4 key

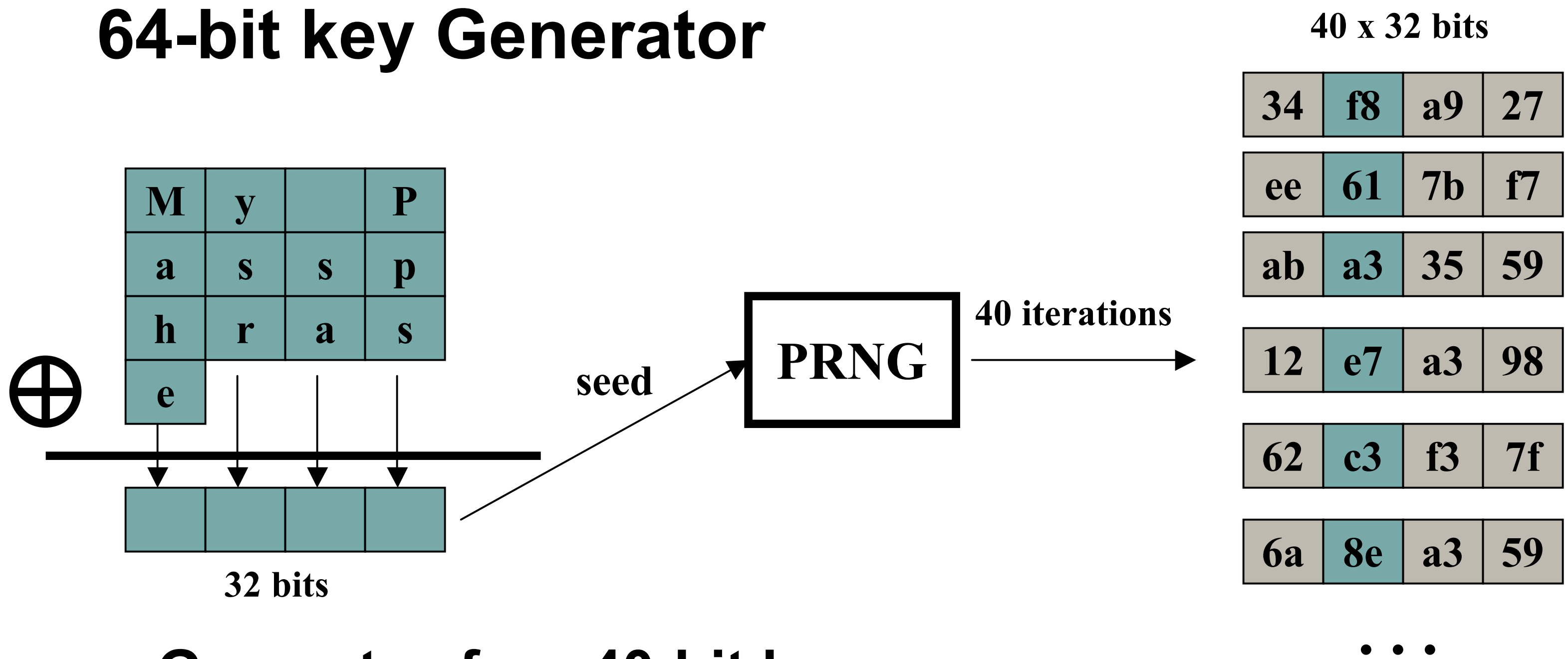
WEP Keying

- **Keys are manually distributed**
- **Keys are statically configured**
  - Implications: often infrequently changed and easy to remember!
- **Four 40-bit keys (or one 104-bit key)**
- **Key values can be directly set as hex data**
- **Key generators provided for convenience**
  - ASCII string is converted into keying material
  - Non-standard but in wide use
  - Different key generators for 64- and 128-bit

Key Entry Example



64-bit key Generator

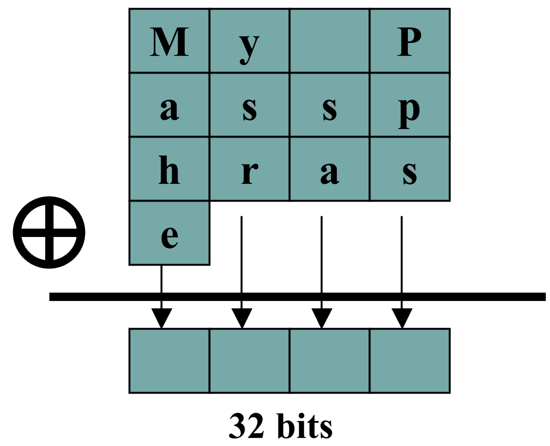


- Generates four 40-bit keys
- ASCII string mapped to 32-bit value with XOR
- Value used as seed to 32-bit linear congruential PRNG
- 40 values generated from PRNG, one byte taken from each 32-bit result

64-bit Generator Flawed!

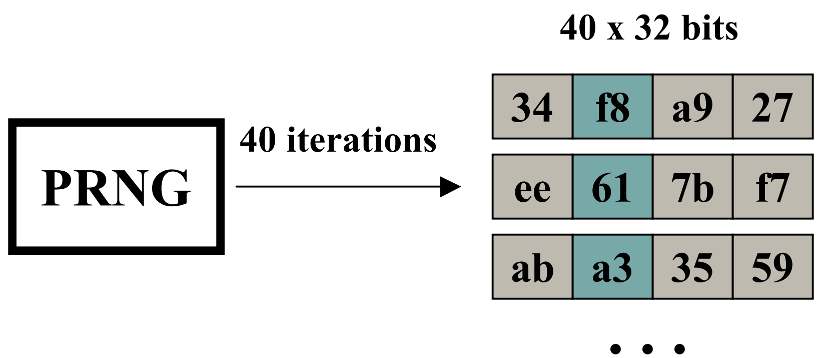
- Ideally should have at least 40-bits of entropy
- Key entropy is reduced in several ways

ASCII Mapping Reduces Entropy



- **ASCII string mapped to 32-bits**
- **XOR operation guarantees four zero bits**
  - Input is ASCII. High bit of each character is always zero
  - XOR of these high bits is also zero
  - Only seeds **00:00:00:00** through **7f:7f:7f:7f** can occur

PRNG Use Reduces Entropy

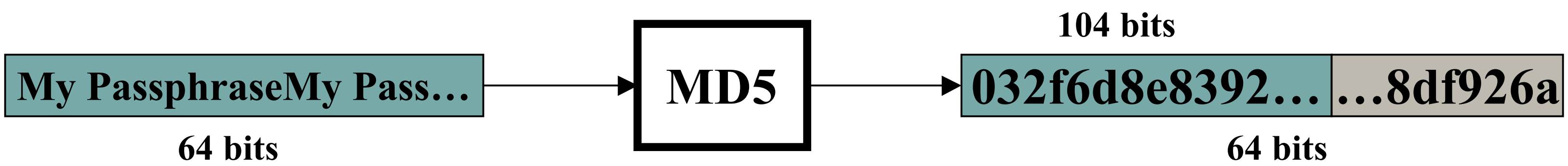


- For each 32-bit output, only bits 16 through 23 are used
- Generator is a linear congruential generator modulo 2^32
  - Low bits are “less random” than higher bits
  - Bit 0 has a cycle length of 2^1, Bit 3 has a cycle length of 2^4, etc..
- The resultant bytes have a cycle length of 2^24
- Only seeds **00:00:00:00** through **00:ff:ff:ff** result in unique keys!

Entropy of 64-bit Generator is 21-bits

- The ASCII folding operation only generates seeds **00:00:00:00** through **7f:7f:7f:7f**
  - High bit of each constituent byte is always zero
- Only seeds **00:00:00:00** through **ff:ff:ff:ff** result in unique keys
- Result: Only 2^21 unique keys generated!
  - Only need to consider seeds **00:00:00:00** through **00:7f:7f:7f** with zero high bits

128-bit Generator

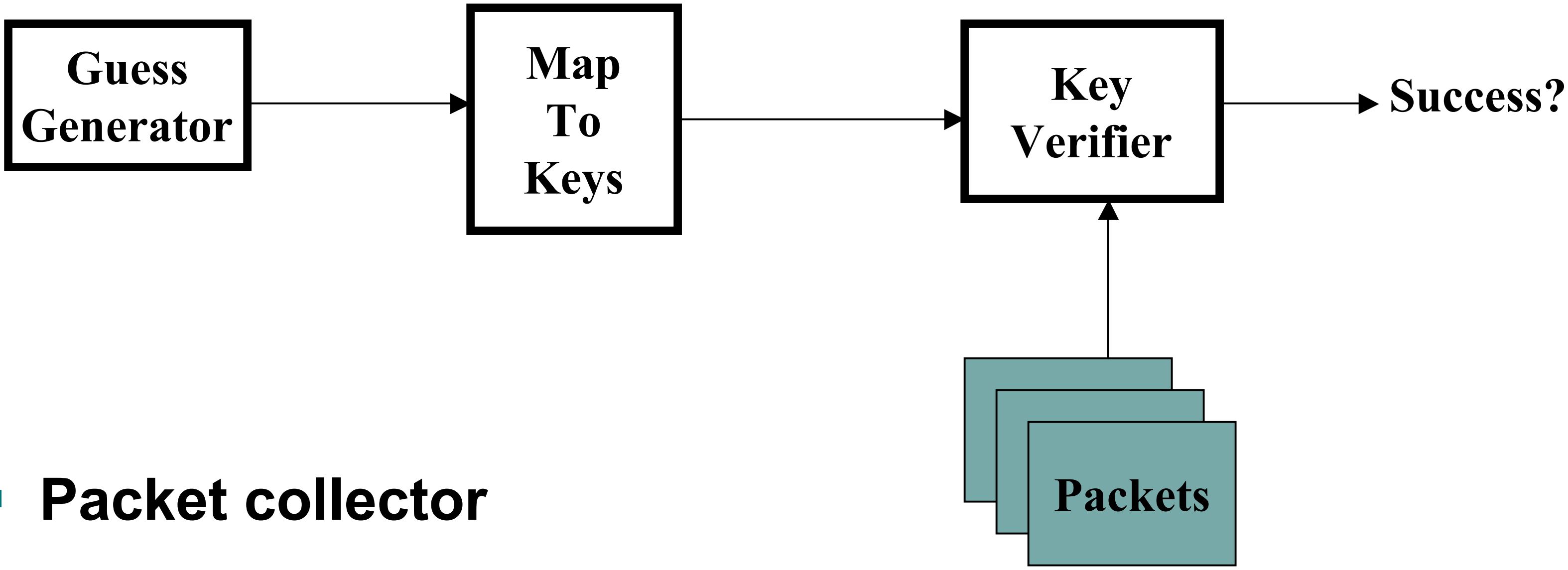


- One 104-bit key is generated
- ASCII string is extended to 64-bytes through repetition
- MD5 of resulting 64-bytes is taken
- 104-bits of output selected
- Key strength relies on the strength of MD5 and of the ASCII string

Designed and Implemented a WEP Cracker

- **Proof of concept: bells and whistles left out**
- **Perform dictionary attack against WEP keys**
  - Find keys generated from a dictionary word
  - Find keys that are ASCII words
    - Consider each of the four 64-bit WEP keys or the single 128-bit WEP key
- **Perform brute force of the weak 64-bit WEP generator**
- **No support for other brute force attacks**

Structure of WEP Cracker



- Packet collector
- Guess Generator
- Mapping guesses to WEP keys
- Key verifier

Packet Collector

- **Collect the appropriate packets needed for guess verification**
  - Collects 802.11 DATA packets
  - Two packets collected
- **Reads from pcap-format file**
  - Simplifies design and allows for off-line cracking
  - Capture utilities such as PrismDump already output to this format

Making Guesses

- **Dictionary attack**
  - Read wordlist from file
  - Lots of room for improvement. For example, rule-based word generation.
- **Brute force of generator**
  - Generate sequential PRNG seeds between **00:00:00:00** and **00:7f:7f:7f**

Mapping Guesses to Keys

- **Direct translation of ASCII to key bytes**
  - Five ASCII bytes mapped to a single 64-bit WEP key
  - Thirteen ASCII bytes mapped to the 128-bit WEP key
  - Truncation of long words, zero-fill for short words
- **Use of the key generator functions**
  - Map ASCII to keys with 64-bit generator
  - Map ASCII to keys with 128-bit generator
  - Map PRNG seeds to keys with 64-bit generator

Key Verification

- **Authentication (Challenge/Response) packets**
  - Easiest to verify
    - Challenge/Responds provides known plaintext
  - Not ideal - Infrequent and optional
- **Data packets**
  - Verify that decrypted packets are well-formed
  - Verify that ICV is correct
  - Inexact: can result in false-positives
    - Verifying against several packets increases assurance

ICV Verification

- Get IV and keynumber from packet
- Form RC4 key from IV+key[keynumber]
- Decrypt payload+ICV
- Recompute ICV and compare
- Probability of false match is 2^-32
  - Matching two packets gives high assurance

Results

- **Proof of concept constructed**
  - Dictionary attack on ASCII keys and 64- and 128-bit key generators
  - Brute force of 64-bit generator
- **Performance on PIII/500MHz laptop**
  - Brute force of 64-bit generator in 35 seconds, 60,000 guesses/second
  - 60,000 guesses/second against 64-bit ASCII keys
  - 45,000 guesses/second against 128-bit generated keys
  - 55,000 guesses/second against 128-bit ASCII keys

Brute Force of Keys

- **Brute force of 40-bit keys is not practical**
  - About 210 days on my laptop
  - ~100 machines could perform attack in reasonable time
  - Better attacks exist
- **Brute force 104-bit keys is not feasible**
  - 10^19 years

Implications

- **64-bit generator should not be used**
- **If ASCII keys or generated keys are used, string should be well chosen**
  - Use similar guidelines as when choosing a login password
- **Random 40-bit keys have reasonable strength**
- **Well chosen 104-bit keys, generated or not, are strong**

Related work – Bad News

- **Ian Goldberg et al and Jesse Walker**
  - WEP encryption is fundamentally flawed
  - Attack times on the order of a few days
- **Bill Arbaugh et al**
  - WEP authentication can be performed without knowing the key
  - Extended Goldberg's attacks against WEP encryption – easier to perform
- **Places upper limit on cracking efforts – 1-2 days**

That's All Folks…

- tnewsham@stake.com
- Source code provided on CD or at http://www.lava.net/~newsham/wlan/
- Source code is Public Domain
- Questions?