BB84

Quantum *Protected* Cryptography

A.C.

August 9, 2021

Outline

EM Waves

- Light is paired waves in electric and magnetic fields

EM Waves

- Light is paired waves in electric and magnetic fields
- General wave equation

$$\nabla^2\Psi = \frac{\partial^2\Psi}{\partial t}$$

EM Waves

- Light is paired waves in electric and magnetic fields
- General wave equation

$$\nabla^2\Psi = \frac{\partial^2\Psi}{\partial t}$$

- Maxwell's equations

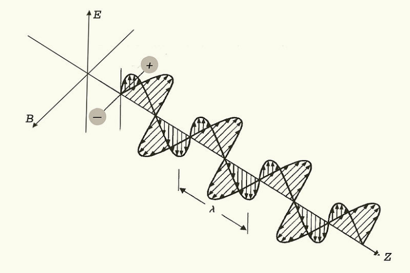
$$\oint_A EdA = \oint_A BdA = 0$$

$$\oint_C Edl = -\int_A \frac{\partial B}{\partial t} dA$$

$$\oint_C Bdl = \mu_0\epsilon \int_A \frac{\partial E}{\partial t} dA$$

Linearly Polarized Light

- Useful simplified model: assume harmonic, planar waves.
  - Between Fourier transforms and other superpositions this covers a surprising amount of ground.
  - Call the orientation of the E-wave the light's *polarization*



Polarizers

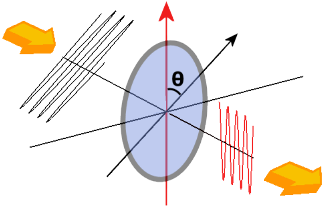
- It is possible to create materials which filter light according to its polarization

Polarizers

- It is possible to create materials which filter light according to its polarization
- All light passing through a polarizer is dimmed according to the alignment of its polarization with the polarizer.

Polarizers

- It is possible to create materials which filter light according to its polarization
- All light passing through a polarizer is dimmed according to the alignment of its polarization with the polarizer.
- Ideal polarizers follow Malus's law: $I = I_0 cos^2\theta$



Quantizing

- EM radiation is transmitted in quanta called *photons*

Quantizing

- EM radiation is transmitted in quanta called *photons*
- Still have a polarization due to their wave aspect
  - or possibly a superposition of different polarizations ...

Quantizing

- EM radiation is transmitted in quanta called *photons*
- Still have a polarization due to their wave aspect
  - or possibly a superposition of different polarizations ...
- Cannot be dimmed – quanta are indivisible.

Quantizing

- EM radiation is transmitted in quanta called *photons*
- Still have a polarization due to their wave aspect
  - or possibly a superposition of different polarizations . . .
- Cannot be dimmed – quanta are indivisible.
- Instead polarizers absorb with probability $\sin^2\theta$
  - or transmit with probability $\cos^2\theta$

No Cloning Theorem

- Observing a quantum entity, including photons, collapses it into a single state.

No Cloning Theorem

- Observing a quantum entity, including photons, collapses it into a single state.
- Impossible to perfectly capture and reconstruct a photon of unknown polarization
  - Approximate cloning *is* possible, e.g. via stimulated emission
  - General upper bound on cloning fidelity of $\frac{5}{6}$
  - Fidelity bound for the case we'll care about of ≈ 0.8535

Qubit Bases

- Photon measurement is deterministic at $\theta = 0$ and $\theta = \frac{\pi}{2}$

Qubit Bases

- Photon measurement is deterministic at $\theta = 0$ and $\theta = \frac{\pi}{2}$
- Map vertically aligned light to $|1\rangle$, horizontally to $|0\rangle$
  - Verticality is purely convention
  - Call the decision of which way is up our *measurement basis*

Qubit Bases

- Photon measurement is deterministic at $\theta = 0$ and $\theta = \frac{\pi}{2}$
- Map vertically aligned light to $|1\rangle$, horizontally to $|0\rangle$
  - Verticality is purely convention
  - Call the decision of which way is up our *measurement basis*
- What happens if sender and receiver use different bases?
  - i.e., there's a phase offset of ϕ between their verticals

Qubit Bases

- Photon measurement is deterministic at $\theta = 0$ and $\theta = \frac{\pi}{2}$
- Map vertically aligned light to $|1\rangle$, horizontally to $|0\rangle$
  - Verticality is purely convention
  - Call the decision of which way is up our *measurement basis*
- What happens if sender and receiver use different bases?
  - i.e., there's a phase offset of ϕ between their verticals
- Bit flip with probability $\sin^2 \phi$, by Malus' law.

The Protocol

1. Alice and Bob publicly agree on two bases (rect, diag) with a $\frac{\pi}{4}$ offset between them

The Protocol

1. Alice and Bob publicly agree on two bases (rect, diag) with a $\frac{\pi}{4}$ offset between them
2. Alice creates a random sequence of bits

The Protocol

1. Alice and Bob publicly agree on two bases (rect, diag) with a $\frac{\pi}{4}$ offset between them
2. Alice creates a random sequence of bits
3. Alice randomly rect- or diag-encodes each bit, transmits to Bob

The Protocol

1. Alice and Bob publicly agree on two bases (rect, diag) with a $\frac{\pi}{4}$ offset between them
2. Alice creates a random sequence of bits
3. Alice randomly rect- or diag-encodes each bit, transmits to Bob
4. Bob randomly rect- or diag-decodes each photon

The Protocol

1. Alice and Bob publicly agree on two bases (rect, diag) with a $\frac{\pi}{4}$ offset between them
2. Alice creates a random sequence of bits
3. Alice randomly rect- or diag-encodes each bit, transmits to Bob
4. Bob randomly rect- or diag-decodes each photon
5. Alice and Bob publicly announce their basis choices, drop bits where their bases disagree.

The Protocol

1. Alice and Bob publicly agree on two bases (rect, diag) with a $\frac{\pi}{4}$ offset between them
2. Alice creates a random sequence of bits
3. Alice randomly rect- or diag-encodes each bit, transmits to Bob
4. Bob randomly rect- or diag-decodes each photon
5. Alice and Bob publicly announce their basis choices, drop bits where their bases disagree.
6. Alice and Bob publicly compare a random subset of the undropped bits.

The Protocol

1. Alice and Bob publicly agree on two bases (rect, diag) with a $\frac{\pi}{4}$ offset between them
2. Alice creates a random sequence of bits
3. Alice randomly rect- or diag-encodes each bit, transmits to Bob
4. Bob randomly rect- or diag-decodes each photon
5. Alice and Bob publicly announce their basis choices, drop bits where their bases disagree.
6. Alice and Bob publicly compare a random subset of the undropped bits.
7. If they observe a high rate of corruption, they start over.

The Protocol

1. Alice and Bob publicly agree on two bases (rect, diag) with a $\frac{\pi}{4}$ offset between them
2. Alice creates a random sequence of bits
3. Alice randomly rect- or diag-encodes each bit, transmits to Bob
4. Bob randomly rect- or diag-decodes each photon
5. Alice and Bob publicly announce their basis choices, drop bits where their bases disagree.
6. Alice and Bob publicly compare a random subset of the undropped bits.
7. If they observe a high rate of corruption, they start over.
8. Otherwise, the undropped and unannounced bits are now a shared secret.

Protocol in Pictures

Alice:	0	0	1	1	0	1	0	1

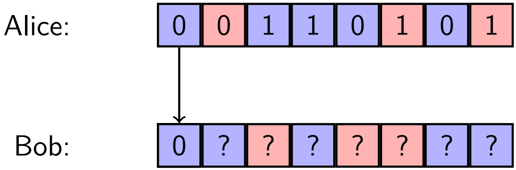
Alice selects random bits, bases

Protocol in Pictures

Alice:	0	0	1	1	0	1	0	1
Bob:	?	?	?	?	?	?	?	?

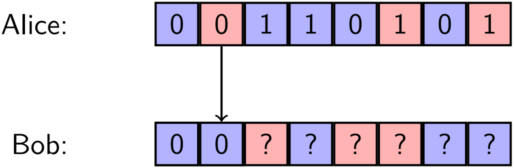
Bob selects random bases

Protocol in Pictures



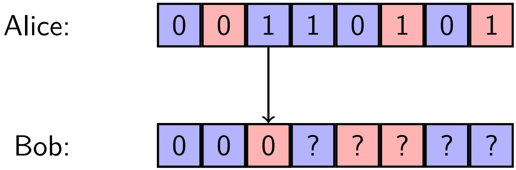
Bases agree, measurement succeeds

Protocol in Pictures



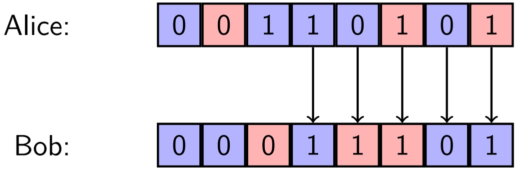
Bases disagree, measurement *may* fail

Protocol in Pictures



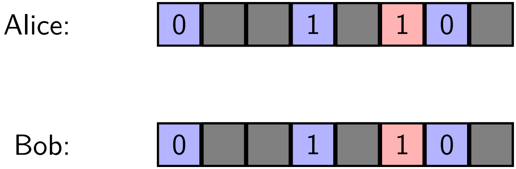
Bases disagree, measurement *may* fail

Protocol in Pictures



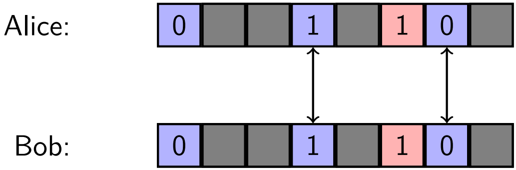
Etc...

Protocol in Pictures



Sift out mismatched bases

Protocol in Pictures

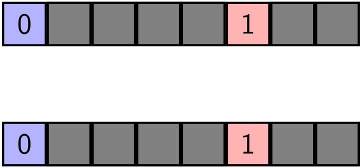


Sample bits to detect Eve

Protocol in Pictures

Alice:

Bob:



Redact sampled bits

BB84 Intuition

- Rectilinear and diagonal bases chosen to maximize corruption
  - Decoding in the wrong basis yields a 50% chance of corruption.

BB84 Intuition

- Rectilinear and diagonal bases chosen to maximize corruption
  - Decoding in the wrong basis yields a 50% chance of corruption.
- If Eve intercepts a photon and forwards it on to Bob, she'll likely corrupt it.
  - 50% chance she's in the wrong basis
  - 50% chance of corruption from mis-encoding.
  - 25% Bob gets a corrupted bit (assuming no intelligent cloning).

BB84 Intuition

- Rectilinear and diagonal bases chosen to maximize corruption
  - Decoding in the wrong basis yields a 50% chance of corruption.
- If Eve intercepts a photon and forwards it on to Bob, she'll likely corrupt it.
  - 50% chance she's in the wrong basis
  - 50% chance of corruption from mis-encoding.
  - 25% Bob gets a corrupted bit (assuming no intelligent cloning).
- Can't stop Eve, but can *detect* her.

Outline

Benign QBER

- Sending and receiving individual quanta is *hard*

Benign QBER

- Sending and receiving individual quanta is *hard*
- Even if Eve isn't interfering, some of our bits will be scrambled.

Benign QBER

- Sending and receiving individual quanta is *hard*
- Even if Eve isn't interfering, some of our bits will be scrambled.
- Secrets which are mostly the same are not nearly as useful as secrets which are exactly the same.

Benign QBER

- Sending and receiving individual quanta is *hard*
- Even if Eve isn't interfering, some of our bits will be scrambled.
- Secrets which are mostly the same are not nearly as useful as secrets which are exactly the same.
- We need to somehow rectify the differences between what Alice sent and Bob received.

Single Error Detection

- Start with 4 data bits d_1, d_2, d_3, d_4.
  - How can we detect a lone bit flip?

Single Error Detection

- Start with 4 data bits d_1, d_2, d_3, d_4.
  - How can we detect a lone bit flip?
- Idea, add parity bit at the end: $p_1 := d_1 \oplus d_2 \oplus d_3 \oplus d_4$

Single Error Detection

- Start with 4 data bits d_1, d_2, d_3, d_4.
  - How can we detect a lone bit flip?
- Idea, add parity bit at the end: $p_1 := d_1 \oplus d_2 \oplus d_3 \oplus d_4$
- If we retrieve our code word $(d_1, d_2, d_3, d_4, p_1)^T$ in the future and it has odd parity, at least one bit must have been flipped.

Single Error Detection

- Start with 4 data bits d_1, d_2, d_3, d_4.
  - How can we detect a lone bit flip?
- Idea, add parity bit at the end: $p_1 := d_1 \oplus d_2 \oplus d_3 \oplus d_4$
- If we retrieve our code word $(d_1, d_2, d_3, d_4, p_1)^T$ in the future and it has odd parity, at least one bit must have been flipped.
- Can write as a check matrix:

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Single Error Correction

- If we want to correct an error, we need to know *which* bit was flipped.

Single Error Correction

- If we want to correct an error, we need to know *which* bit was flipped.
- Let's add more parity bits, but
  - Each parity bit will only cover some bits in the code word.
  - We'll be careful to make sure that every bit is covered by a unique combination of parity bits

Single Error Correction

- If we want to correct an error, we need to know *which* bit was flipped.
- Let's add more parity bits, but
  - Each parity bit will only cover some bits in the code word.
  - We'll be careful to make sure that every bit is covered by a unique combination of parity bits
- Hamming solved this for us in 1950 using powers of two
  - Code word: $(p_1, p_2, d_1, p_3, d_2, d_3, d_4)^T$
  - Check matrix:

$$H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Full SECDED

- Glue a total parity bit onto a Hamming SEC scheme
  - No errors $\implies$ all parity checks report even
  - Single error $\implies$ total parity reports odd, SEC parities indicate location
  - Double error $\implies$ total parity reports even, SEC parities indicate “location”

Full SECDED

- Glue a total parity bit onto a Hamming SEC scheme
  - No errors $\implies$ all parity checks report even
  - Single error $\implies$ total parity reports odd, SEC parities indicate location
  - Double error $\implies$ total parity reports even, SEC parities indicate “location”
- Commonly known as Hamming(8,4).
  - Code word: $(p_1, p_2, d_1, p_3, d_2, d_3, d_4, p_{\text{total}})^T$
  - Check matrix:

$$H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Winnow I

- What if we just pretended our sifted qubits were SECDED code words?

Winnow I

- What if we just pretended our sifted qubits were SECDED code words?
- They're not, so if Alice has a block a, then Ha will almost always want to "correct" something.
  - And for Bob, Hb will behave similarly

Winnow I

- What if we just pretended our sifted qubits were SECDED code words?
- They're not, so if Alice has a block a, then Ha will almost always want to "correct" something.
  - And for Bob, Hb will behave similarly
- $a \oplus b \approx 0 \implies H(a \oplus b) = Ha \oplus Hb \approx 0$

Winnow I

- What if we just pretended our sifted qubits were SECDED code words?
- They're not, so if Alice has a block a, then Ha will almost always want to "correct" something.
  - And for Bob, Hb will behave similarly
- $a \oplus b \approx 0 \implies H(a \oplus b) = Ha \oplus Hb \approx 0$
- What if we compare Ha and Hb, have Bob apply SECDED using $Ha \oplus Hb$?
  - If even errors, do nothing
  - If odd errors, probably $d(a, b') = d(a, b) - 1$

Winnow I

- What if we just pretended our sifted qubits were SECDED code words?
- They're not, so if Alice has a block a, then Ha will almost always want to "correct" something.
  - And for Bob, Hb will behave similarly
- $a \oplus b \approx 0 \implies H(a \oplus b) = Ha \oplus Hb \approx 0$
- What if we compare Ha and Hb, have Bob apply SECDED using $Ha \oplus Hb$?
  - If even errors, do nothing
  - If odd errors, probably $d(a, b') = d(a, b) - 1$
- We can reconcile errors by exchanging Hamming syndromes!

Winnow II

1. Alice and Bob break their bits into SECDED code words

Winnow II

1. Alice and Bob break their bits into SECDED code words
2. Alice and Bob exchange total parity for each codeword
    2.1 blocks whose total parities agree are ignored going forward

Winnow II

1. Alice and Bob break their bits into SECDED code words
2. Alice and Bob exchange total parity for each codeword
   2.1 blocks whose total parities agree are ignored going forward
3. Alice sends Bob *Ha* for each remaining block

Winnow II

1. Alice and Bob break their bits into SECDED code words
2. Alice and Bob exchange total parity for each codeword
    2.1 blocks whose total parities agree are ignored going forward
3. Alice sends Bob Ha for each remaining block
4. Bob applies SEC using $Ha \oplus Hb$

Winnow II

1. Alice and Bob break their bits into SECDED code words
2. Alice and Bob exchange total parity for each codeword
   2.1 blocks whose total parities agree are ignored going forward
3. Alice sends Bob Ha for each remaining block
4. Bob applies SEC using $Ha \oplus Hb$
5. Alice and Bob discard all announced parity bits
   5.1 Keeps Eve from gleaning info from our error correction process.

Winnow II

1. Alice and Bob break their bits into SECDED code words
2. Alice and Bob exchange total parity for each codeword
   2.1 blocks whose total parities agree are ignored going forward
3. Alice sends Bob *Ha* for each remaining block
4. Bob applies SEC using $Ha \oplus Hb$
5. Alice and Bob discard all announced parity bits
   5.1 Keeps Eve from gleaning info from our error correction process.
6. Alice and Bob shuffle their remaining bits using the same seed.
   6.1 Stops an even number of errors from “sticking” in a code word.

Winnow II

1. Alice and Bob break their bits into SECDED code words
2. Alice and Bob exchange total parity for each codeword
    2.1 blocks whose total parities agree are ignored going forward
3. Alice sends Bob *Ha* for each remaining block
4. Bob applies SEC using $Ha \oplus Hb$
5. Alice and Bob discard all announced parity bits
    5.1 Keeps Eve from gleaning info from our error correction process.
6. Alice and Bob shuffle their remaining bits using the same seed.
    6.1 Stops an even number of errors from "sticking" in a code word.
7. Go back to (1) until convinced that the probability of surviving errors is negligible.

Bit Leakage

- Real systems have some amount of legitimate error

Bit Leakage

- Real systems have some amount of legitimate error
- Paranoia requires that we treat all errors as due to eavesdropping

Bit Leakage

- Real systems have some amount of legitimate error
- Paranoia requires that we treat all errors as due to eavesdropping
- *Eve is allowed to have partial information on our secret!*
  - Or we can just never succeed in negotiating a secret

Bit Leakage

- Real systems have some amount of legitimate error
- Paranoia requires that we treat all errors as due to eavesdropping
- *Eve is allowed to have partial information on our secret!*
  - Or we can just never succeed in negotiating a secret
- Need to do two things
  - Estimate how many bits of information Eve has
  - Somehow scrub those bits out of our secret

Parameter Estimation I

- Whenever Eve performs an intercept/resend, she risks creating an error
  - $\text{QBER} \sim \frac{1}{n} \cdot \text{Bin}(n, p)$

Parameter Estimation I

- Whenever Eve performs an intercept/resend, she risks creating an error
  - QBER $\sim \frac{1}{n} \cdot \mathrm{Bin}(n, p)$
  - Theoretical bound of $2\sqrt{2}$ bits of information per bit flipped (in the long run)

Parameter Estimation I

- Whenever Eve performs an intercept/resend, she risks creating an error
  - QBER $\sim \frac{1}{n} \cdot \text{Bin}(n, p)$
  - Theoretical bound of $2\sqrt{2}$ bits of information per bit flipped (in the long run)
  - If we can estimate p we can determine how many bits Eve has

Parameter Estimation I

- Whenever Eve performs an intercept/resend, she risks creating an error
  - $\text{QBER} \sim \frac{1}{n} \cdot \text{Bin}(n, p)$
  - Theoretical bound of $2\sqrt{2}$ bits of information per bit flipped (in the long run)
  - If we can estimate p we can determine how many bits Eve has
- Estimate $\hat{p} = \text{QBER}$?
  - Too easy to have $\hat{p} < p$

Parameter Estimation I

- Whenever Eve performs an intercept/resend, she risks creating an error
  - $\text{QBER} \sim \frac{1}{n} \cdot \text{Bin}(n, p)$
  - Theoretical bound of $2\sqrt{2}$ bits of information per bit flipped (in the long run)
  - If we can estimate p we can determine how many bits Eve has
- Estimate $\hat{p} = \text{QBER}$?
  - Too easy to have $\hat{p} < p$
- Bayes rule to pick $\hat{p}$ s.t. $P(p > \hat{p}|\text{QBER}) < \epsilon$?
  - No sensible prior to put on p

Parameter Estimation II

- Treat M, our bits to scrub, as an random variable $f(X)$

Parameter Estimation II

- Treat M, our bits to scrub, as an random variable $f(X)$
- If $P(M < 2\sqrt{2} \cdot p) < \epsilon$, we win
  - Sounds an awful lot like a confidence interval

Parameter Estimation II

- Treat M, our bits to scrub, as an random variable $f(X)$
- If $P(M < 2\sqrt{2} \cdot p) < \epsilon$, we win
  - Sounds an awful lot like a confidence interval
- Calculating CIs for a binomial distribution on the fly can be cumbersome
  - Common to pick m, p_{crit} in advance
  - Always trim off m bits, bail if $QBER > p_{\text{crit}}$

Mopping up the Leak

- Soooo, can we just hash our secret from n bits down to m?

Mopping up the Leak

- Soooo, can we just hash our secret from n bits down to m?
- Yes, if we:
  - use a universal hash family
  - trim an additional $2\log\left(\frac{1}{\epsilon}\right)$ bits
  - choose our hash function randomly, although it *can* be public

Mopping up the Leak

- Soooo, can we just hash our secret from n bits down to m?
- Yes, if we:
  - use a universal hash family
  - trim an additional $2\log\left(\frac{1}{\epsilon}\right)$ bits
  - choose our hash function randomly, although it *can* be public
- This is a direct result of the leftover hash lemma

Mopping up the Leak

- Soooo, can we just hash our secret from n bits down to m?
- Yes, if we:
  - use a universal hash family
  - trim an additional $2\log\left(\frac{1}{\epsilon}\right)$ bits
  - choose our hash function randomly, although it *can* be public
- This is a direct result of the leftover hash lemma
- Approach is obvious:
  1. Alice publicly announces a random seed
  2. Alice and Bob both use the seed to hash their secrets down

MITM

- Trivial attack: Eve plops herself between Alice and Bob

MITM

- Trivial attack: Eve plops herself between Alice and Bob
- Alice and Bob wind up with different secrets, but Eve knows both and they only ever talk to her.

MITM

- Trivial attack: Eve plops herself between Alice and Bob
- Alice and Bob wind up with different secrets, but Eve knows both and they only ever talk to her.
- Need some sort of authentication scheme
  - Impossible without some sort of bootstrap secret!
  - But we can still use the scheme to stretch the bootstrap secret, e.g. to allow for one-time pads in the data plane.

MITM

- Trivial attack: Eve plops herself between Alice and Bob
- Alice and Bob wind up with different secrets, but Eve knows both and they only ever talk to her.
- Need some sort of authentication scheme
  - Impossible without some sort of bootstrap secret!
  - But we can still use the scheme to stretch the bootstrap secret, e.g. to allow for one-time pads in the data plane.
- Our goal is unconditional security, so a standard HMAC won't cut it.

One Time MAC

1. Use l bits of bootstrap secret to choose a member of a universal hash family.

One Time MAC

1. Use l bits of bootstrap secret to choose a member of a universal hash family.
2. Hash the message to authenticated into an m-bit message tag.

One Time MAC

1. Use l bits of bootstrap secret to choose a member of a universal hash family.
2. Hash the message to authenticated into an m-bit message tag.
3. Encrypt the tag with a one-time pad.

One Time MAC

1. Use l bits of bootstrap secret to choose a member of a universal hash family.
2. Hash the message to authenticated into an m-bit message tag.
3. Encrypt the tag with a one-time pad.
4. Discard the m bits used for the OTP.

One Time MAC Intuitions

- Eve can't manufacture a hashes for messages, doesn't know the hash function.
  - Otherwise, she could backward engineer the pad used to encrypt legitimate MACs

One Time MAC Intuitions

- Eve can't manufacture a hashes for messages, doesn't know the hash function.
  - Otherwise, she could backward engineer the pad used to encrypt legitimate MACs
- The hash value is encrypted, so she can't manufacture a hash collision.
  - Universal hash function means she can't even make intelligent guesses about the hash value.

One Time MAC Intuitions

- Eve can't manufacture a hashes for messages, doesn't know the hash function.
  - Otherwise, she could backward engineer the pad used to encrypt legitimate MACs
- The hash value is encrypted, so she can't manufacture a hash collision.
  - Universal hash function means she can't even make intelligent guesses about the hash value.
- Best she can do is guess the tag
  - Succeeds with probability $\frac{1}{2^m}$
  - Places a tradeoff between key expenditure and probability that Eve gets lucky.

PNS

- Emitting a single photon is bloody hard

PNS

- Emitting a single photon is bloody hard
- Most experimental setups work by dimming a laser pulse down to ≈ 1 photon
  - Possible for a pulse to include multiple photons.

PNS

- Emitting a single photon is bloody hard
- Most experimental setups work by dimming a laser pulse down to ≈ 1 photon
  - Possible for a pulse to include multiple photons.
- Eve could sniff multi-photon pulses without corrupting them

PNS

- Emitting a single photon is bloody hard
- Most experimental setups work by dimming a laser pulse down to ≈ 1 photon
  - Possible for a pulse to include multiple photons.
- Eve could sniff multi-photon pulses without corrupting them
- Can also block single-photon pulses to stop Alice/Bob from using them in the secret.
  - Bob expects a lot of transmissions to get lost, this won't necessarily set off alarm bells

PNS

- Emitting a single photon is bloody hard
- Most experimental setups work by dimming a laser pulse down to ≈ 1 photon
  - Possible for a pulse to include multiple photons.
- Eve could sniff multi-photon pulses without corrupting them
- Can also block single-photon pulses to stop Alice/Bob from using them in the secret.
  - Bob expects a lot of transmissions to get lost, this won't necessarily set off alarm bells
- Attack is commonly known as photon number splitting

Decoy States

- Alice uses two (or more) signal sources S and S'
  - S is the real photon source, averages μ photons per pulse
  - S' is a decoy, with $\mu' > \mu$
  - Upshot: S sends more single photon pulses than S'

Decoy States

- Alice uses two (or more) signal sources S and S'
  - S is the real photon source, averages μ photons per pulse
  - S' is a decoy, with $\mu' > \mu$
  - Upshot: S sends more single photon pulses than S'
- Impossible for Eve to tell whether a given pulse came from S or S'

Decoy States

- Alice uses two (or more) signal sources S and S'
  - S is the real photon source, averages μ photons per pulse
  - S' is a decoy, with $\mu' > \mu$
  - Upshot: S sends more single photon pulses than S'
- Impossible for Eve to tell whether a given pulse came from S or S'
- By blocking single photon pulses, Eve will affect the transmission rate for S more than for S'.

Decoy States

- Alice uses two (or more) signal sources S and S'
  - S is the real photon source, averages μ photons per pulse
  - S' is a decoy, with $\mu' > \mu$
  - Upshot: S sends more single photon pulses than S'
- Impossible for Eve to tell whether a given pulse came from S or S'
- By blocking single photon pulses, Eve will affect the transmission rate for S more than for S'.
- Alice and Bob can compare empirical transmission loss for S and S' at the end.
  - If they differ, they conclude Eve did something naughty.

TL;DR

1. Send/receive a stream of quanta in random bases.
2. Throw away the bits where bases disagreed.
3. Reconcile keys using (adaptations of) standard error correction schemes.
4. Hash your keys down according to pessimistic estimates of how much information Eve could have.
5. Make sure to authenticate all your messages throughout.
6. Profit!

References and Additional Materials

- Presentation source: https://github.com/alan-christopher/bb84/tree/master/edu
- Quantum Cryptography Intro: https://arxiv.org/abs/1002.1237
- Quantum Cloning: https://arxiv.org/abs/quant-ph/0511088
- Privacy Amplification: https://link.springer.com/article/10.1007/BF00191318
- Winnow: https://arxiv.org/abs/quant-ph/0203096
- Decoy states: https://arxiv.org/abs/quant-ph/0211153
- A QKD implementation: https://arxiv.org/abs/1603.08387

Included Works

- Transverse Wave Image
- Polarizer Image

Questions?