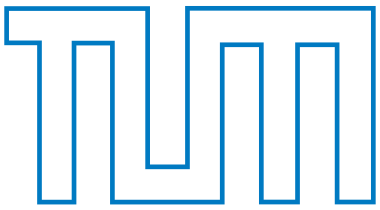


Network Security

Summary

Author: Thomas Pettinger

2017–03–02

Network Security
Technische Universität München

Table of Contents

1 Introduction

1.1 Attacks and Attack Detection

Attacks can have different impacts on the target. Disruptive attacks try to fully deny the service (DoS) of the victim whereas degrading ones only occupy parts of the resources. A DoS attack can also executed distributed, a so called DDoS Attackers might also try to gain confidential data or control the target system. Port scans can be used to gain information about the network topology, operating systems and applications or application versions.

To be able to tell if a system is under attack, different measures can be taken at different points in the system.
Host intrusion detection systems (HIDS) are located on the host system. This enables easy detection using information available on the potential victim system but it has to be present on every system (expensive deployment) and the attack actually reaches the victim and is not detected in advance. Network intrusion detection systems (NIDS) lay on the network layer which enables the detection of attacks before they reach the host.
One of the detection methods available is knowledge-based detection. Known signatures of attacks are compared to the actual traffic and if the patterns match an alarm is raised. This only detects known attacks though. To improve this shortcoming, anomaly detection in traffic, protocol or applciaiton behavior can be used. Anomalies can be detected with different metrics in mind. A very simple one might be the number of request, but this does not take legitimate change in traffic into account. A better approach is using cumulative sums which are low if the average is small or whenever only small amounts of values are large but grows if the amount of large values grows in a certain point in time. The disadvantage of anomaly detection though is that oftentimes the rate of false-positives is high.
Detecting attacks is not easy and network monitory often comes at a cost. It is important though especially in large systems when the attack surface grows. The challenge is to find a good compromise between security and performance.

1.2 Attacker Model and Locations

We generally assume the attacker to be (in) the network. They can perform any active or passive attack but cannot break cryptographic primitives. Active attacks are attacks where some influence is measurable e.g. a delay, modifications or replays whereas a passive attack simply stands for eavesdropping messages. This model is called the Dolev-Yao attacker model.

Attackers can be located on different parts of the network and depending on this location different attacks are possible. If the attacker is close to you they are able to perform active attacks like message modifications on you. This can be circumvented by communicating over a secure tunnel though. If the attacker is close to your servers, timing attacks are possible where attackers can measure how long certain operations take to break into the system. The last possibility is that the attacker is somewhere in the Internet. Since the end user has no control over how packets are routed, attackers can modify the path they take for example.

1.3 Security Goals

Data Integrity No improper or unauthorized change of data

Confidentiality Concealment of information

Availability Services should be available and function correctly

Authenticity Entity is who she claims to be

Accountability Identify the entity responsible for any communication event

Controlled Access Only authorized entities can access certain services or information

1.4 Threads

We define a thread in a communication as any possible event or sequence of actions that might lead to a violation of one or more security goals. The actual realization of a thread is then called attack.

Different threads are possible:

- **Masquerade**: An entity claims to be another entity (also called "impersonation")
- **Eavesdropping**: An entity reads information it is not intended to read
- **Loss or Modification of (transmitted) Information**: Data is being altered or destroyed
- **Denial of Communication Acts (Repudiation)**: An entity falsely denies its participation in a communication act
- **Forgery of Information**: An entity creates new information in the name of another entity
- **Sabotage/Denial of Service**: Any action that aims to reduce the availability and / or correct functioning of services or systems
- **Authorization Violation:**: An entity uses a service or resources it is not intended to use

2 Language-theoretic Security

Communication protocols define the procedure and the format of exchanged messages. If two communication partners speak the same protocol it is not necessarily given that they have the same understanding though. Different problems may arise which result in some guidelines to avoid them:

Full input recognition Programmers usually assume well formed input where in reality the input is controlled by the attacker. For this reason every input should be checked if it fulfills the expectations of valid inputs.

Only Type 2 or 3 of Chomsky Hierarchy grammars for inputs Do not define turing-complete protocols because recognition of the input and testing the equivalence of implementations is undecidable.

Grammar	Language	Recognized by
Type 3	Regular	Finite state automaton
Type 2	Context-free	Pushdown automaton
Type 1	Context-sensitive	Some weird stuff
Type 0	recursively enumerable	Turing machine

Figure 1: Chomsky Hierarchy of Languages

Reduce computing power Reduce the computing power exposed to the outside. Computing power that is not there can not be exploited. This also includes defining protocols only in type 2 or 3 languages since parsing types 0 and 1 requires large or potentially unlimited amounts of computing.

Same interpretation of messages All participants of a communication should interpret messages the same way. For this parsers have to be equivalent which is only decidable for type 2 and 3.

3 Firewalls and Security Policies

We define a system as secure if, started in an allowed state, always stays in states that are allowed.

3.1 The three Security Components

- **Requirements** define security goals
- **Policy** are rules to implement the requirements
- **Mechanisms** enforce the policy

3.2 Network Firewalls

Firewalls provide **controlled access** at the network level to resources. They are usually deployed where a protected subnet is connected to a less trusted network like the internet (c.f. Figure 2). From the firewall's point of view, there are incoming and outgoing packages on all interfaces. These can be filtered using two different strategies.
Whitelisting, which is considered as the best practice, has the default behavior of denying all packets and allowing only explicitly permitted ones whereas **blacklisting** allows everything except blocked packets.

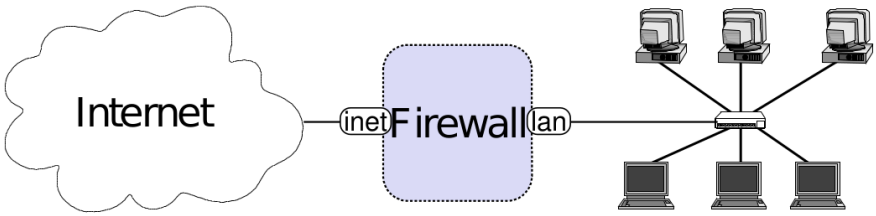


Figure 2: Firewall Placement

Configuring Firewalls

Firewalls are configured with rulesets which are traversed sequentially until a matching one is found. Those rules are composed of a matching condition and an action. Possible actions are for example accept, drop, reject or log. The matching conditions are a bit more complex. They include the incoming interface, several layer 2 to 4 packet fields (srcMAC, dstMAC, srcIP, dstIP, prot, srcPort, dstPort, flags,...), states (in case of stateful matching) and other relevant conditions.

Stateful Firewalls

Stateful Firewalls keep states of connections with the help of IP-5-Tuples (srcIP, dstIP, protocol, srcPort, dstPort). A new state is generated when a packet with a new tuple arrives which transitions from NEW to ESTABLISHED state. It is usually advisable to put more frequently used rules to the top of the firewall, i.e. matching established connections (new connections are rarer) and to check that ports are above the well defined port range (≥ 1024). Figure 3 shows an example of a stateful firewall configuration.

Rule	Iface	Src IP	Dst IP	Protocol	Src Port	Dst Port	State	Action
A	*	*	*	*	*	*	Est.	Accept
B	inet	external	webserver	TCP	> 1023	80	New	Accept
C	lan	internal	external	TCP	> 1023	80	New	Accept
D	lan	internal	external	UDP	> 1023	53	New	Accept
E	*	*	*	*	*	*	*	Drop

Figure 3: Stateful Firewall Configuration

Stateless Firewalls

Stateless firewalls do not generate states for incoming connections but only operate on single packets since keeping states is expensive and needs fast memory. For this reason lookup times are in $\mathcal{O}(\#rules)$ which, for large $\#rules$ is slower than stateful filtering ($\mathcal{O}(1)$). For small $\#rules$ stateless filtering is faster though due to the lack of memory writes. It is also more complex to configure which makes the approach more error prone. An example for a stateless ruleset is shown in Figure 4.

Rule	Iface	Src IP	Dst IP	Protocol	Src Port	Dst Port	Ack	Action
B_1	inet	external	webserver	TCP	> 1023	80	*	Accept
B_2	lan	webserver	external	TCP	80	> 1023	Yes	Accept
C_1	lan	internal	external	TCP	> 1023	80	*	Accept
C_2	inet	external	internal	TCP	80	> 1023	Yes	Accept
D_1	lan	internal	external	UDP	> 1023	53	-	Accept
D_2	inet	external	internal	UDP	53	> 1023	-	Accept
E	*	*	*	*	*	*	*	Drop

Figure 4: Stateless Firewall Configuration

Stateless firewalls usually look at the ACK flag of TCP connection to determine approximately if connections are new or established. When a SYN/ACK packet is sent as first packet of a connection, the firewall will pass it but the host will drop it if implement properly.

Spoofing Protection

Spoofing is the forgery of IP addresses by filling the source IP field with another IP than one's own. Spoofing protection then allows only IP addresses that belong to you on outgoing connections. For incoming traffic this is more difficult since we can not determine where the packet actually came from so we are only able to block our and special purpose IPs.

Common Errors

- How is your firewall management interface reachable?
- What is allowed over the Internet?
- IPv4 and IPv6?
- Outbound rule ANY? (c.f. spoofing)
- Policy's vs. Firewalls understanding of Inbound and Outbound?
- Shadowing: unreachable firewall rules

What Firewalls cannot do

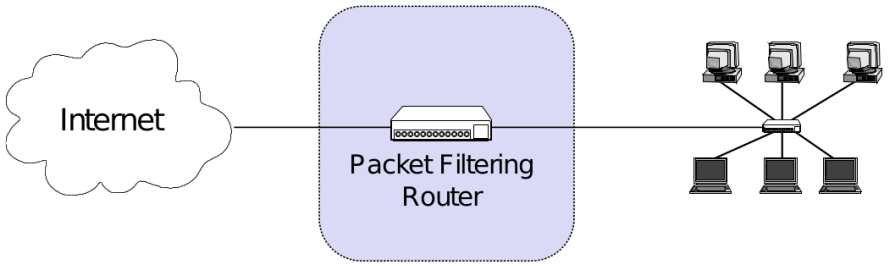
- can't protect against malicious insiders
- can't protect against connections that don't go through it
- can't protect against completely new threats
- can't fully protect against viruses
- does not perform cryptographic operations, e.g. message authentication
- can't set itself up correctly

Bastion Hosts

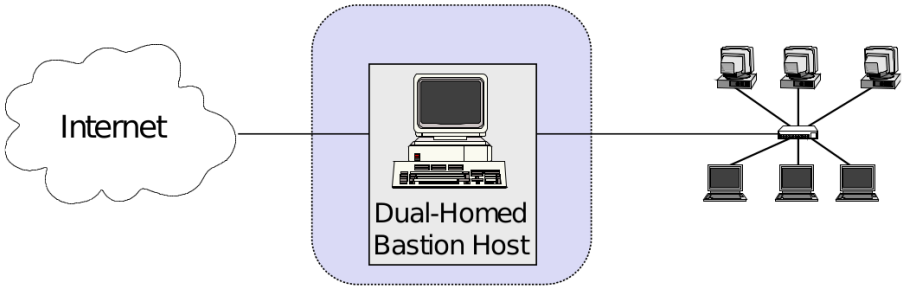
A bastion host is a host that is more exposed to the hosts of an external network than the other hosts of the network it protects. When configuring, one should keep in mind that it might get compromised so do things like disabling SSH password login, do not allow it to sniff internal traffic or disable user accounts.

Firewall Architectures

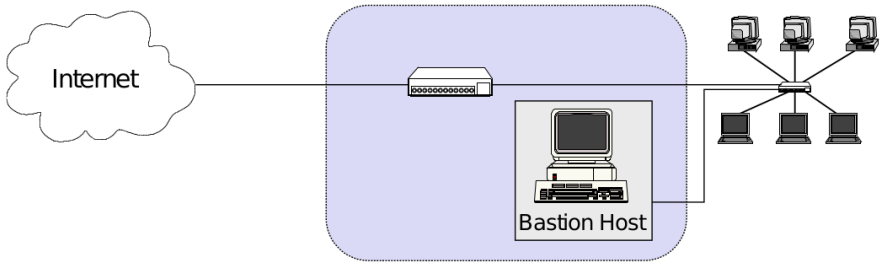
Simple Packet Filter Architecture Packet filtering is done through a filtering router or firewall with two interfaces.



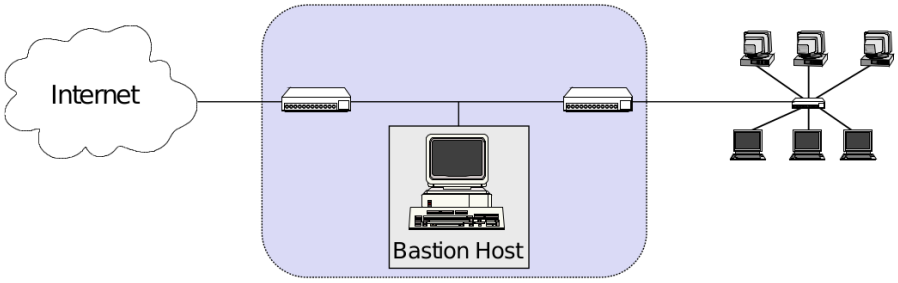
Dual-Homed Host Architecture The bastion host is part of two networks and is firewall and application proxy. Disadvantage: bastion host is bottleneck



Screened Host Architecture The bastion host is proxy, located in the internal network and thus protected by the firewall.



Screened Subnet Architecture - DMZ A demilitarized zone (DMZ) is configured hosting the bastion host (proxy) and publicly accessible servers. The second packet filter is an additional protection measurement in case the DMZ gets compromised.



4 TCP SYN Cookies

4.1 TCP SYN Flood Attack

During an TCP SYN Flood Attack an attacker sends large amounts of packets with spoofed source addresses to the victim. This fills up their connection table with half open connections (sequence numbers) which results in legitimate users not being able to establish new TCP connections. A solution for this are TCP SYN cookies.

4.2 TCP SYN Cookies

TCP SYN Cookies are particularly chosen initial sequence numbers $\alpha = h(S, K_{SYN})$ where K is a secret key, S_{SYN} the source address of the SYN packet and h a cryptographic hash function. On arrival of the ACK message, Bob calculates α again and checks if the ACK number is correct $(\alpha + 1)$. This process is shown in Figure 5.

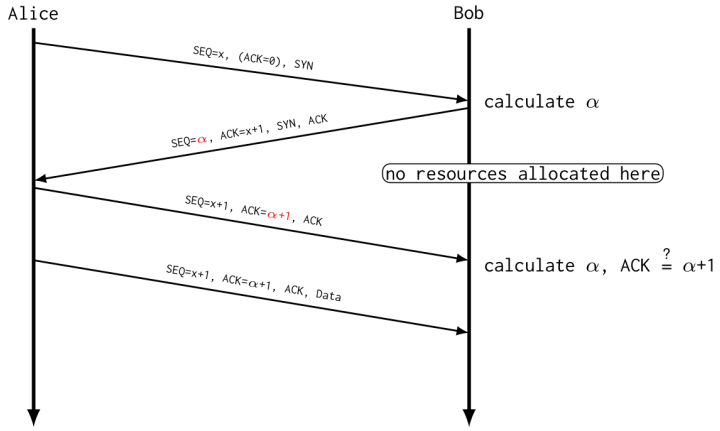


Figure 5: TCP SYN Cookies

Pros

- No resource allocation after SYN packets
- Client does not have to be aware of the server using SYN cookies
- No changes in the TCP protocol necessary

Cons

- Calculating α may be CPU consuming (Linux implementation: CPU local with high cashing efficiency)
- TCP options (like large windows size) cannot be negotiated (Linux implementation: window size hacked into cookie, SYN cookies only enabled if a threshold of connections is exceeded)
- Efficient implementations might be vulnerable to cryptoanalysis (Linux implementation: SHA used, counter updated every minute)

5 Symmetric Cryptography

In symmetric encryption two communication partners Alice and Bob share a secret key that is used for encryption and decryption. It ensures only confidentiality, not integrity or authenticity.

We use the following terminology:

- Key k
- Plaintext $m = Dec_k(c)$
- Ciphertext $c = Enc_k(m)$
- $Dec_k(Enc_k(m)) = m$

5.1 One-Time-Pad (OTP): A Perfect Cypher

For the OTP a perfectly random bitstream opt with the length of the message to encode is needed. Then the encryption operation is defined as $Enc_{otp}(m) = m \oplus otp$ and the decryption operation as $Dec_{otp}(c) = c \oplus otp$
For the OTP to be perfectly secure, the key must only be used once which is impractical in the real world where we usually want $length(k) \ll length(m)$ and thus reusable keys.

5.2 Security of Cyphers

Kerckhoff's principle states that the cipher method must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience. Thus only the key needs to be secret.
It is advisable to use only standardized ciphers from libraries and RTFM (read the fucking manual) because implementing own ciphers has a lot of pitfalls and is more often or almost always done wrong. Furthermore encryption does not imply that the system is secure, often integrity and authenticity is more important than confidentiality. And also do not forget key management.

5.3 Attacking Symmetric Ciphers

The goals of attacks on ciphers is to learn something about m given a c. Getting any information about k from an attack is also considered a successful one.

Possible attack scenarios are:

- Ciphertext-only-attack: attacker knows c
- Known-plaintext-attack: For a fixed k, the attacker got a pair (m, c) and tries to learn something about other ciphertexts
- Chosen-plaintext and chosen-ciphertext attack: similar to previous attack, but attacker can chose m or c freely

A cipher is secure if the best known attack is brute-forcing all keys.

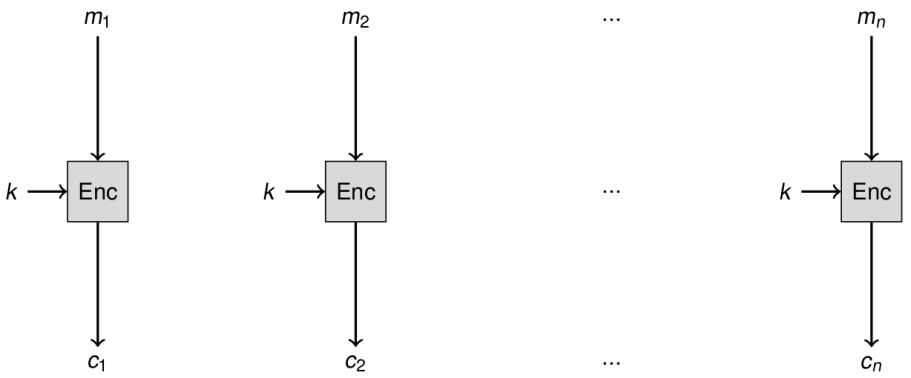
5.4 Block and Stream Ciphers

A block cipher encrypts and decrypts inputs of length n to outputs of length n ($\Rightarrow$ block length n). A steam cipher on the other hand generates a random bitstream with arbitrary length, called keystream, that is xored to the plain text to encrypt and decrypt.
The most advisable cipher to use is probably the AES block cipher. It is well tested and proven to be (mostly) secure and hardware supported what makes it quite fast ($> 2GB/s$ with HW support, $200Mbit/s$ without).

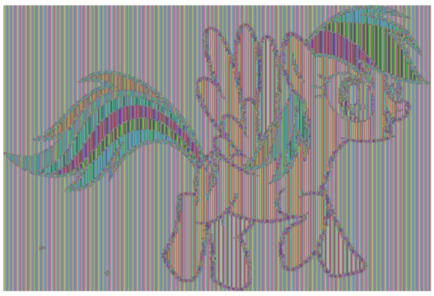
5.5 Modes of Encryption

Modes of encryption are necessary to handle messages of variable length with block ciphers. The plaintext therefore is split into parts of length equal to the cipher block length. If the last block is shorter than the block length, we add padding.

5.6 Electronic Code Book Mode (ECB)



In ECB, every plaintext block is encrypted with the unmodified key as input. The problem thereby is that identical plaintext blocks result in the same ciphertext blocks.



5.7 Cipher Block Chaining Mode (CBC)

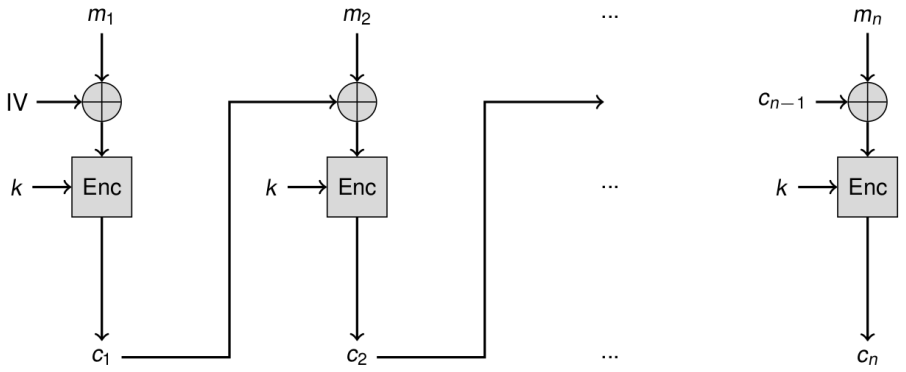


Figure 6: CBC Encrypt

CBC uses the ciphertext from the previous block xored with the current message block and the unmodified key k as inputs of the encryption function $c_i = Enc_k(c_{i-1} \oplus m_i)$. For the first message, where no previous cipher block is available, an initialization vector (IV) is used which may be sent in plaintext and is fresh for every massage (or packet if the message is split across multiple packets).
Compared to ECB, the advantage is that equal plaintext blocks/messages are not encrypted to the same ciphertext blocks/messages.

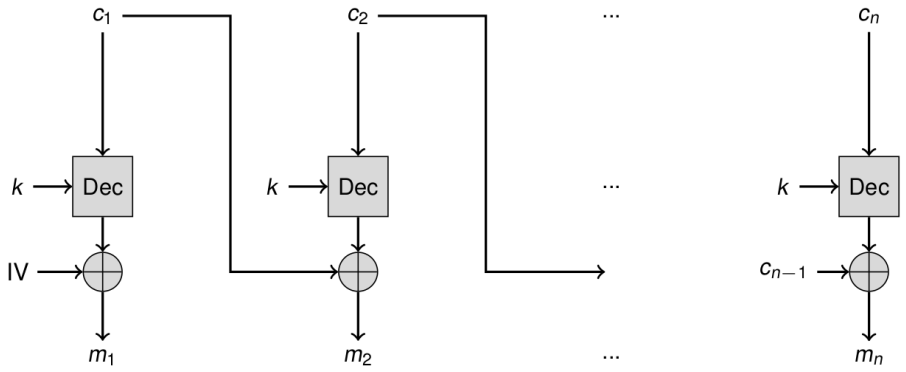


Figure 7: CBC Decrypt

Decryption is defined by $m_i = c_{i-1} \oplus Dec_k(c_i)$.

5.8 Output Feedback Mode (OFB)

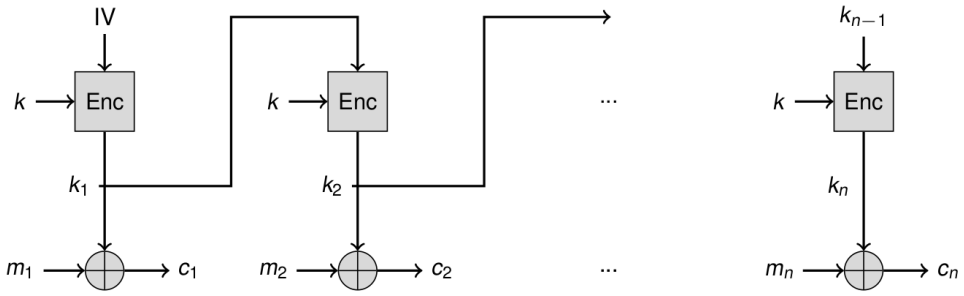


Figure 8: OFB Encrypt

OFB transforms a block cipher into a stream cipher. In step i, it takes cipher key k_{i-1} form the previous step and encrypts it with the key k. This cipher key k_i is then xored with the corresponding plaintext block m_i to calculate the ciphertext c_i. In the first step an IV is used as k_{i-1}.

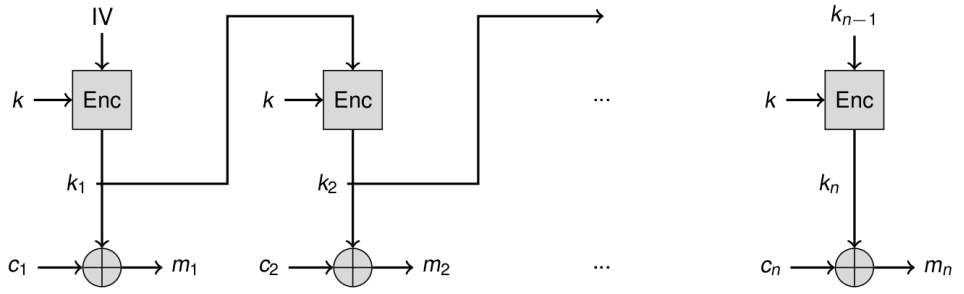


Figure 9: OFB Decrypt

Decryption is the same procedure except that the key stream is xored with the cipher text blocks.

5.9 Counter Mode (CTR)

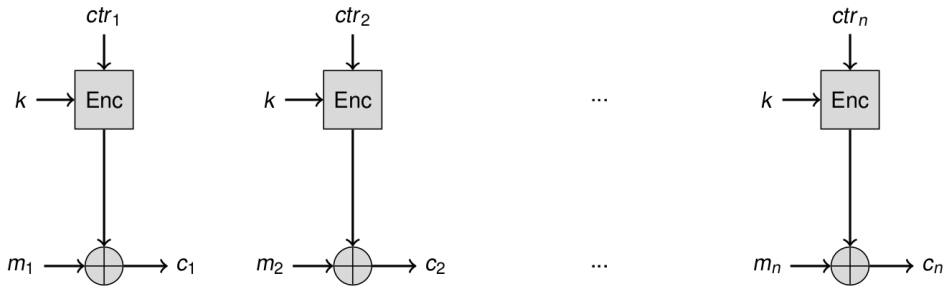


Figure 10: CTR Encrypt

In step i, CTR encrypts a counter $ctr_i = IV||i$ with key k and xors the result with the plaintext block m_i. Like OFB, CTR also transforms block ciphers into stream ciphers.

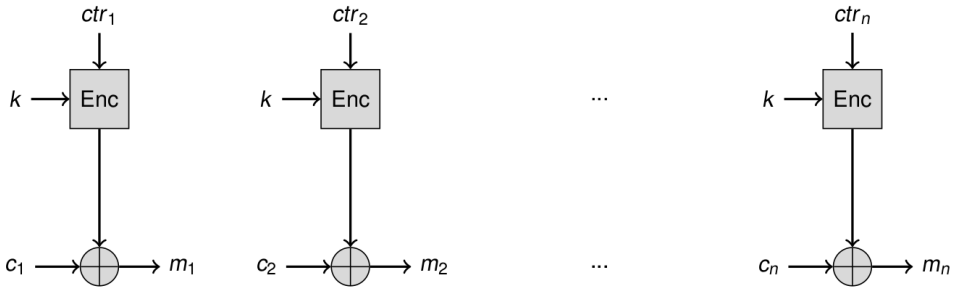


Figure 11: CTR Decrypt

Decryption is done by $m_i = Enc_k(ctr_i) \oplus c_i$.