2026/11/11, 2026 보안 기초 세미나

암호학과 네트워크 보안

-1장 보안 개요, 2장 암호 수학-

서 진 우(jinwoo@pel.sejong.ac.kr)

세종대학교 프로토콜공학연구실

목 차

- 보충
- 보안 개요
- 암호 수학

목 차

- 보충
- 보안 개요
- 암호 수학

보충

- 정수 연산(Integer Arithmetic)
  - 선형 디오판투스 방정식(Liner Diophantine Equations)
    - 해 계산
      - 특수 해
        1. d로 방정식 양변을 나누어, $a_1x + b_1y = c1$으로 만듦
        2. 확장 유클리드 알고리즘을 사용하여, $a_1s + b_1t = 1$을 만족하는 s와 t 계산
        3. 특수 해는 $x_0 = (c/d)s,\ y_0 = (c/d)t$로 계산 가능
      - 일반 해
        - 특수 해를 찾아낸 뒤, 일반 해는 다음과 같음
        - $x = x_0 + k(b/d),\ y = y_0 - k(b/d)$, 이때 k는 정수
    - 활용
      - 다른 값들을 갖는 물체의 여러 조합을 탐색할 경우 응용
        - 100달러 수표를 20달러와 5달러 짜리로 현금화 할 경우, $200x + 5y = 100$을 계산하여 조합 탐색
          - $d = \gcd(20, 5) = 5$이고 $5|100$이므로 무수히 많은 해 존재
          - 특수 해는 $x_0 = 020 = 0, y_0 = 120 = 20$
          - 일반 해는 $(0, 20), (1, 16), (2, 12), (3, 8), (4, 4), (5, 0)$

보충

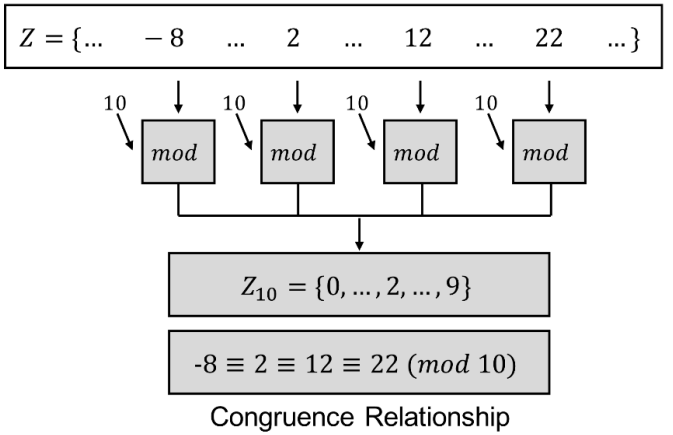
- 모듈로 연산(Modular Arithmetic)
  - 잉여류(Residue Class) $[a]$

*최소 잉여 집합(Set of Least Residue Modulon) Z_n
모듈로 연산을 통해 나온 나머지들의 집합

    - 정의
      - n으로 나눈 나머지가 같은 정수들의 집합
        - $[a] = \{x : x \equiv a(mod\ n)\}$
    - 특징
      - 모듈로 n는 $[0], [1], \dots, [n-1]$의 n개의 잉여류 존재
      - 각 잉여류는 무한히 많은 정수 포함
      - 각 잉여류의 가장 작은 음이 아닌 정수가 그 잉여류를 대표하며, 대푯값들을 모은 집합이 *최소 잉여 집합 Z_n이라고 표현
    - 예시
      - $n = 3$일 경우
        - $[0] = \{\dots, -6, -3, 0, 3, 6, \dots\}, [1] = \{\dots, -5, -2, 1, 4, 7, \dots\}, \ [2] = \{\dots, -4, -1, 2, 5, 8, \dots\}$

보충

- 모듈로 연산(Modular Arithmetic)
  - 합동(Congruence)
    - 정의
      - 모듈로 n 연산 시, 나머지 같은 수들에 대한 관계
    - 특징
      - 합동 연산자(Congruence Operator) ≡ 사용
      - Z의 무한한 원소는 Z_n의 한 원소로 대응 가능
        - e.g., $2\ mod\ 10 = 2,\ 12\ mod\ 10 = 2,\ 22\ mod\ 10 = 2$일 때, 2, 12, 22은 $mod\ 10$에 대해 합동이므로 $2 \equiv 12 \equiv 22\ (mod\ 10)$로 표기 가능
    - 합동 연산자와 등식 연산자의 차이점
      - 등식 연산자
        - Z의 원소를 Z의 원소로 대응(Mapping)
        - 일-대-일(One-to-One)
      - 합동 연산자
        - Z의 원소에서 Z_n의 원소로 대응
        - 다-대-일(Many-to-One)



Congruence Relationship

목 차

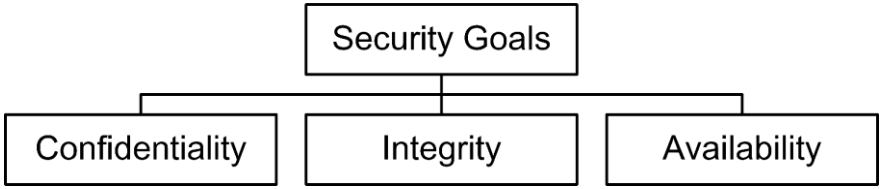
- 보안 개요
- 암호 수학

보안 개요

- 보호(Protection)
  - 시스템 내부에서 권한이 없는 사용자나 프로세스의 자원 접근을 제어하는 메커니즘
- 보안(Security)
  - 시스템을 위협으로부터 보호하고 기밀성, 무결성, 가용성을 보장하기 위한 포괄적인 대책

보안 개요

- 보안 목표(Security Goals)
  - 개요
    - 정보 자산을 안전하게 보호하기 위해 최종적으로 달성하고자 하는 핵심 기준
  - CIA Triad
    - 기밀성(Confidentiality)
      - 인가된 사람/프로세스/시스템만이 알 필요성에 근거하여 시스템에 접근해야 한다는 원칙
    - 무결성(Integrity)
      - 정보는 고의적인, 비인가된, 우연한 변경으로 보호되어야 한다는 원칙
    - 가용성(Availability)
      - 정보는 인가된 사용자가 필요로 하는 시점에 접근 가능해야 한다는 원칙



보안 개요

- 공격(Attack) (1/5)
  - 구분
    - 보안 목표에 대한 공격
      - 기밀성을 위협하는 공격
      - 무결성을 위협하는 공격
      - 가용성을 위협하는 공격
    - 시스템에 미치는 영향에 따른 공격
      - 소극적(Passive) 공격
      - 적극적(Active) 공격

보안 개요

- 공격(Attack) (2/5)
  - 보안 목표에 대한 공격
    - 기밀성을 위협하는 공격
      - 스누핑(Snooping)
        - 비인가 접근을 통해 데이터를 무단으로 열람하거나 탈취하는 공격
        - e.g., 암호화되지 않은 와이파이 환경이나 *키로거 등을 통해 발생 가능
      - 트래픽 분석(Traffic Analysis)
        - 통신 내용이 암호화되어 있어도 통신 패턴과 *메타데이터를 분석하여 정보를 유추하는 공격
        - e.g., 특정 금융 기관 서버와 특정 기업과 주기적으로 대용량 패킷이 오가는 패턴을 분석하여, 중요한 금융 거래 발생을 유추
      - 사회 공학(Social Engineering)
        - 사람을 속여 기밀 정보를 직접 제공하도록 유도하는 공격
        - e.g., 피싱 메일로 사용자의 계정 정보를 입력하도록 유도

*키로거(Keylogger)
키보드로 입력하는 타이핑을 몰래 기록하여 탈취하는 악성 소프트웨어 또는 하드웨어 장치
*메타데이터(Metadata)
통신 내용 자체가 아닌 송수신자, 통신 시간, 데이터 크기 등의 부가 정보

보안 개요

- 공격(Attack) (3/5)
  - 보안 목표에 대한 공격
    - 무결성을 위협하는 공격
      - 변경(Modification)
        - 전송 중이거나 저장된 데이터를 무단으로 수정하는 공격
        - e.g., 공격자가 송금 금액을 10만 원에서 100만 원으로 변경
      - 가장(Masquerading)
        - 다른 사용자나 시스템인 것처럼 신분을 위장하는 공격
        - e.g., 공격자가 다른 사용자의 계정으로 로그인하여 정상 사용자로 위장
      - 재전송(Replaying)
        - 과거에 전송된 유효한 데이터를 가로채 다시 전송하는 공격
        - e.g., 공격자가 이전에 캡처한 결제 승인 메시지를 재전송하여 중복 결제를 유도
      - 부인(Repudiation)
        - 자신이 수행한 행위나 전송한 메시지에 대해 사실이 아니라고 부인하는 공격
        - e.g., 사용자가 온라인으로 송금한 뒤, 자신은 송금을 요청한 적 없다고 주장

보안 개요

- 공격(Attack) (4/5)
  - 보안 목표에 대한 공격
    - 가용성을 위협하는 공격
      - 서비스 거부(Denial of Service)
        - 시스템에 과도한 요청을 보내거나 자원을 고갈시켜 정상적인 서비스 제공을 방해하는 공격
        - e.g., 공격자가 서버에 대량의 요청을 보내 정상 사용자가 웹 사이트에 접속하지 못하게 하는 공격
      - 분산 서비스 거부(Distributed Denial of Service)
        - 여러 시스템에서 동시에 공격하여 서비스를 제공을 방해하는 공격
        - e.g., 다수의 감염된 컴퓨터로 웹 서버 공격
      - 자원 고갈 공격(Resource Exhaustion)
        - CPU, 메모리, 저장 공간 등의 자원을 소진 시켜 시스템의 정상 작동을 방해하는 공격
        - e.g., 메모리를 과도하게 사용하도록 유도하여 서버가 응답하지 못하게 하는 공격
      - 서비스 파괴(Service Disruption)
        - 시스템이나 서비스의 핵심 구성 요소를 손상시켜 이용할 수 없게 하는 공격
        - e.g., 서버의 중요 파일을 삭제하여 서비스가 중단되도록 하는 공격

보안 개요

- 공격(Attack) (5/5)
  - 시스템에 미치는 영향 따른 공격
    - 소극적(Passive) 공격
      - 정보를 획득하거나 통신을 관찰하며, 데이터나 시스템을 직접 변경하지 않는 공격
        - 스누핑, 트래픽 분석
    - 적극적(Active) 공격
      - 데이터를 변경·위조하거나 시스템의 정상적인 동작을 방해하는 공격
        - 사회 공학
        - 변경, 가장, 재전송, 부인
        - 서비스 거부, 분산 서비스 거부, 자원 고갈 공격, 서비스 파괴

보안 개요

- 서비스와 메커니즘 (1/6)
  - 보안 서비스 (1/2)
    - 데이터 기밀성(Data Confidentiality)
      - 노출 공격으로부터 데이터의 내용을 보호하는 서비스
      - e.g., *AES를 이용해 데이터를 암호화하여 비인가자가 내용을 확인하지 못하도록 함
    - 데이터 무결성(Data Integrity)
      - 공격자의 데이터 변경, 삽입, 삭제 및 재전송 등으로부터 데이터를 보호하는 서비스
      - e.g., *MAC을 검증하여 전송된 메시지가 변경되었는지 확인 함
    - 인증(Authentication)
      - 사용자의 신원 또는 데이터의 출처가 정당한지 확인하는 서비스
      - e.g., 서버 인증서를 검증하여 접속한 서버가 정당한 서버인지 확인 함

*AES (Advanced Encryption Standard)
대칭키 암호화 방식의 대표적인 알고리즘으로, 데이터를 일정한 크기의 블록 단위로 암호화
*MAC (Message Authentication Code)
데이터의 무결성과 발신자 인증을 위해 메시지에 덧붙이는 짧은 데이터 조각

보안 개요

- 서비스와 메커니즘 (2/6)
  - 보안 서비스 (2/2)
    - 부인봉쇄(Nonrepudiation)
      - 송신자나 수신자가 특정 행위를 수행한 사실을 나중에 부인하지 못하도록 증거를 제공하는 서비스
      - e.g., 전자문서에 디지털 서명을 적용하여 송신자가 문서를 보낸 사실에 대한 증거를 제공 함
    - 접근 제어(Access Control)
      - 비인가자의 시스템 및 데이터 접근을 제한하는 서비스
      - e.g., 사용자별 접근 권한을 설정하여 관리자만 서버 설정을 변경할 수 있도록 함

보안 개요

- 서비스와 메커니즘 (3/6)
  - 보안 메커니즘 (1/3)
    - 암호화(Encipherment)
      - 데이터를 암호문으로 변환하여 비인가자가 내용을 알 수 없도록 하는 메커니즘
      - e.g., AES를 이용해 파일을 암호화하여 암호키가 없는 사용자가 내용을 읽지 못하도록 함
    - 데이터 무결성(Data Integrity)
      - 데이터의 변경이나 손상 여부를 확인하는 메커니즘
      - e.g., MAC을 검증하여 메시지가 전송 중 변경되었는지 확인 함
    - 디지털 서명(Digital Signature)
      - 송신자가 개인키로 서명을 생성하고, 수신자가 송신자의 공개키로 서명을 검증하는 메커니즘
      - e.g., 전자문서에 개인키로 서명하고 수신자가 공개키로 검증하여 서명과 문서의 무결성을 확인 함

보안 개요

- 서비스와 메커니즘 (4/6)
  - 보안 메커니즘 (2/3)
    - 인증 교환(Authentication Exchange)
      - 신원을 증명하기 위해 인증 정보를 주고받는 메커니즘
      - e.g., 서버가 인증용 난수를 보내고 사용자가 이에 대한 응답을 생성하여 신원을 증명 함
    - 트래픽 패딩(Traffic Padding)
      - 데이터 스트림에 더미 데이터를 삽입하여 통신 패턴 분석을 어렵게 만드는 메커니즘
      - e.g., 실제 데이터가 없을 때도 더미 데이터를 전송하여 통신 여부와 빈도를 파악하기 어렵게 함
    - 라우팅 제어(Routing Control)
      - 보안상 안전한 경로를 선택하도록 데이터의 전달 경로를 제어하는 메커니즘
      - e.g., 도청 위험이 있는 네트워크 경로를 피해 신뢰할 수 있는 경로로 데이터를 전송 함

보안 개요

- 서비스와 메커니즘 (5/6)
  - 보안 메커니즘 (3/3)
    - 공증(Notarization)
      - 신뢰할 수 있는 제3자가 데이터 교환과 관련된 사실을 확인하거나 증명하는 메커니즘
      - e.g., 신뢰할 수 있는 제3자가 전자문서의 송수신 시각을 기록하여 거래 사실을 증명 함
    - 접근 제어(Access Control)
      - 사용자의 접근 권한을 확인하여 시스템이나 데이터에 대한 접근을 허용하거나 제한하는 메커니즘
      - e.g., 사용자 계정과 권한을 확인하여 허가된 사용자만 특정 파일을 열 수 있도록 함

보안 개요

- 서비스와 메커니즘 (6/6)
  - 서비스와 메커니즘의 관계
    - 서비스는 보안 위협으로부터 보호하기 위해 제공하는 기능 정의
    - 메커니즘은 보안 서비스를 구현하기 위해 사용하는 방법 정의

Security Service	Security Mechanism
데이터 기밀성	암호화, 라우팅 제어
데이터 무결성	암호화, 디지털 서명, 데이터 무결성
인증	암호화, 디지털 서명, 인증 교환
부인방지	디지털 서명, 데이터 무결성, 공증
접근 제어	접근 제어

보안 개요

- 기술 (1/8)
  - 암호(Cryptography) (1/4)
    - 정의
      - 정보를 보호하기 위해 암호화와 복호화, 해싱 등의 기법을 사용하는 기술
    - 종류
      - 대칭-키 암호화(Symmetric-key Encipherment)
      - 비대칭-키 암호화(Asymmetric-key Encipherment)
      - 해싱(Hashing)

보안 개요

- 기술 (2/8)
  - 암호(Cryptography) (2/4)
    - 대칭-키 암호화(Symmetric-key Encipherment)
      - 정의
        - 암호화와 복호화에 동일한 비밀키를 사용하는 암호화 방식
      - 동작 원리
        - 송신자는 비밀키로 평문을 암호화하고, 수신자는 동일한 비밀키로 암호문을 복호화
      - 특징
        - 암호화 및 복호화 속도가 빠름
        - 암호화 프로토콜을 통해 비밀키를 안전하게 공유하고 관리해야 함
          - e.g., Diffie-Hellman
      - 예시
        - AES
        - *DES/3DES

*DES (Data Encryption Standard)
56비트의 키를 이용하는 대칭키 암호시스템으로, 데이터를 64비트 단위의 블록으로 분할하여 순열, 배타적 OR, 회전 등으로 암호화
*Diffie-Hellman
각자 비밀값과 공개된 값을 조합해 교환하고, 양측이 동일한 비밀키를 생성하는 방식

보안 개요

- 기술 (3/8)
  - 암호(Cryptography) (3/4)
    - 비대칭-키 암호화(Asymmetric-key Encipherment)
      - 정의
        - 공개키와 개인키로 구성된 한 쌍의 키를 사용하는 암호 기술
      - 동작 원리
        - 기밀성 확보 시 수신자의 공개키로 암호화하고, 수신자는 개인키로 복호화
        - 전자서명 시 개인키로 서명하고, 공개키로 서명을 검증
      - 특징
        - 공개키를 공개할 수 있어 비밀키를 사전에 공유할 필요가 없음
        - 대칭키 방식보다 일반적으로 연산 속도가 느림
      - 예시
        - *RSA
        - *ECC

*RSA (Rivest-Shamir-Adleman)
큰 소수의 곱을 소인수분해하기 어렵다는 수학적 성질을 기반으로 하는 비대칭키 암호 알고리즘
*ECC (Elliptic Curve Cryptography)
타원 모양의 곡선 그래프에서 특정 점을 반복해서 더하는 규칙을 이용한 공개키 암호화 방식

보안 개요

- 기술 (4/8)
  - 암호(Cryptography) (4/4)
    - 해싱(Hashing)
      - 정의
        - 임의 길이의 입력 데이터를 해시 함수로 변환하여 고정 길이의 해시 값을 생성하는 기술
      - 동작 원리
        - 입력 데이터를 해시 함수에 넣어 해시 값을 생성
      - 특징
        - 입력 데이터이 조금 변경되더라도 해시 값은 달라질 수 있음
        - 해시 값으로부터 원본 데이터를 복원하기 어려움
        - 원본과 수신 데이터의 해시 값을 통해 데이터의 무결성 검증에 활용
        - 암호화와 달리 복호화 과정이 없음
      - 예시
        - *SHA-256
        - *MD5

*SHA-256 (Secure Hash Algorithm 256-bit)
임의 길이의 입력 데이터를 256비트(32바이트)의 고정 길이 해시 값으로 변환하는 암호학적 해시 함수
*MD5 (Message-Digest Algorithm 5)
파일이나 텍스트를 입력받아 128비트길이의 고정된 해시값으로 변환하는 암호화 해시 함수

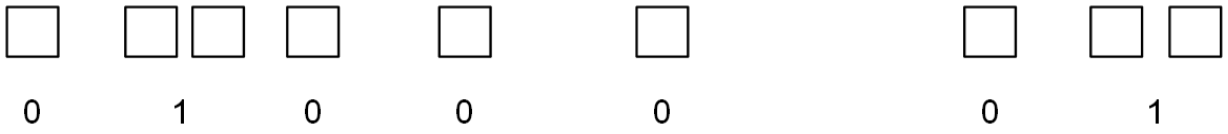
보안 개요

- 기술 (5/8)
  - 스테가노그래피(Steganography) (1/4)
    - 정의
      - 비밀 정보를 이미지, 음성, 영상 등의 일반 데이터에 숨겨 정보의 존재 자체를 감추는 기법
    - 역사상의 활용
      - 고대 그리스에서 노예의 머리를 깎아 두피에 메시지를 새긴 뒤, 머리카락이 자라면 전달하는 방식으로 비밀 정보를 은닉
    - 현대의 사용
      - 텍스트 커버(Text Cover)
      - 이미지 커버(Image Cover)
      - 기타 커버 방식

보안 개요

- 기술 (6/8)
  - 스테가노그래피(Steganography) (2/4)
    - 텍스트 커버(Text Cover)
      - 정의
        - 텍스트 문서의 공백, 글자 위치, 문장 구조 등을 이용해 비밀 정보를 숨기는 방식
      - 특징
        - 구현이 다른 커버 방식에 비해 비교적 간단
        - 삽입할 수 있는 정보의 양이 제한적
        - 문서 편집이나 서식 변경으로 은닉 정보가 손상될 가능성 존재
      - 예시
        - 문장 사이 공백의 개수에 따라 0과 1의 비트 정보를 삽입

This book is mostly about cryptography, not steganography



보안 개요

- 기술 (7/8)

*LSB (Least Significant Bit)
최하위 비트로, 2진수에서 가장 작은 자릿값을 가지는 마지막 비트

  - 스테가노그래피(Steganography) (3/4)
    - 이미지 커버(Image Cover)
      - 정의
        - 이미지의 픽셀 값을 조작하여 비밀 정보를 숨기는 방식
      - 특징
        - *LSB를 주로 활용
        - 픽셀 값의 미세한 변화는 육안으로 식별하기 어려움
        - 이미지 압축이나 크기 변경으로 은닉 정보가 손상될 가능성 존재
      - 예시
        - 픽셀 값의 최하위 비트를 변경하여 비밀 데이터 삽입
        - e.g., $10110010 \rightarrow 10110011$

보안 개요

- 기술 (8/8)
  - 스테가노그래피(Steganography) (4/4)
    - 기타 커버 방식
      - 음성 커버
        - 오디오 신호의 미세한 변화를 이용해 비밀 정보 삽입
      - 영상 커버
        - 프레임의 픽셀 값에 비밀 정보 삽입
      - 네트워크 커버
        - 프로토콜 헤더의 사용되지 않는 필드 등을 이용해 정보 은닉

목 차

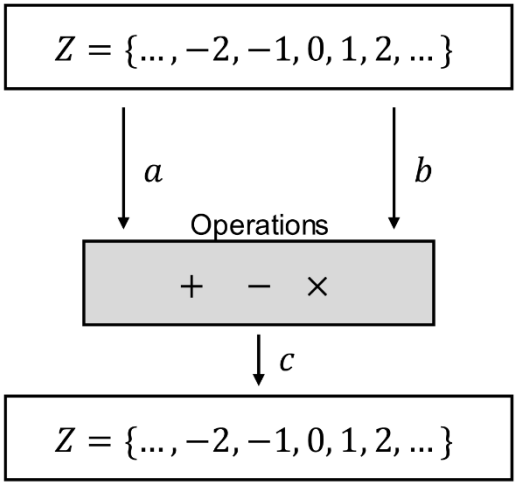
- 보안 개요
- 암호 수학

암호 수학

- 정수 연산(Integer Arithmetic) (1/12)
  - 개요
    - *정수 집합에서의 산술 연산

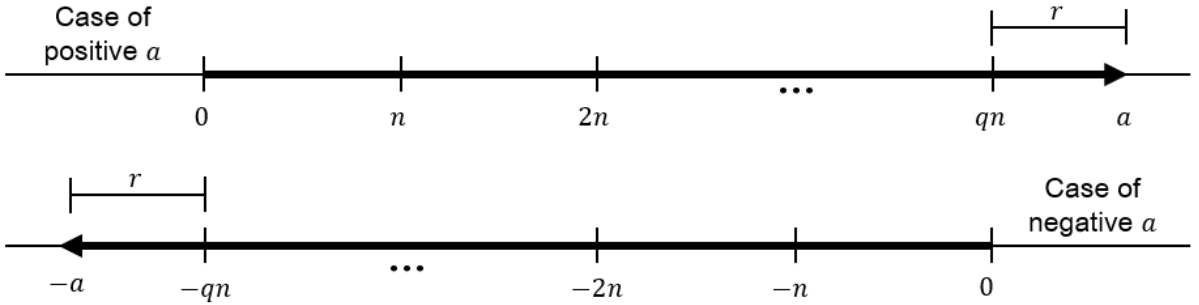
*정수 집합(Set of Integers) Z
음의 무한대에서부터 양의 무한대까지의 모든 정수
($Z = \{\dots, -2, -1, 0, 1, 2, \dots\}$)

  - 2진 연산(Binary Operations)
    - 세 가지 2진 연산 사용
      - 덧셈(Add), 뺄셈(Subtract), 곱셈(Multiply)
        - e.g., $5 + 9 = 14, (-5) + 9 = 4$
        - e.g., $5 - 9 = -4, (-5) - 9 = -14,$
        - e.g., $5 \times 9 = 45, (-5) \times 9 = 45$
    - 두 개의 입력 값에 대하여 하나의 결과 값 산출
      - 두 개의 입력 값은 정수 집합에서 선택
      - 결과 값 또한 정수 집합의 원소

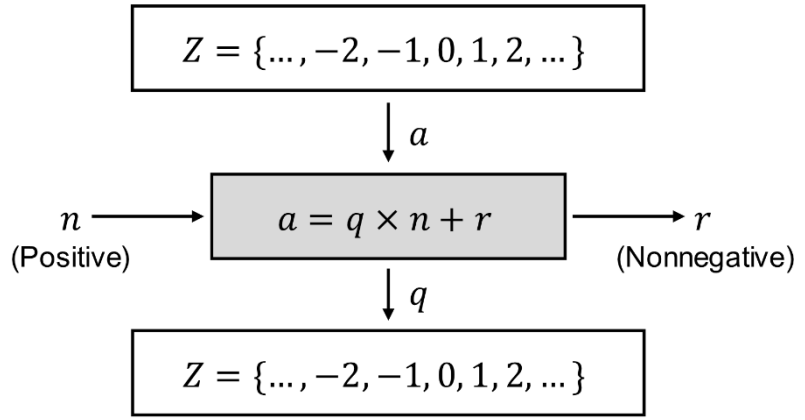


암호 수학

- 정수 연산(Integer Arithmetic) (2/12)
  - 정수의 나눗셈
    - 정수 연산에서 a를 n으로 나누면 q와 r을 얻음
      - $a = q \times n + r$
      - a는 피제수, n은 제수, q는 몫, r은 나머지이며, 모두 정수
    - 제한 사항
      - $n > 0$
      - $r \geq 0$
    - 예시
      - $a = 255, n = 11, q = 23, r = 2$



$$\begin{array}{r|l} & 23 \leftarrow q \\ \hline n \rightarrow 11 & 255 \leftarrow a \\ & 22 \\ \hline & 35 \\ & 33 \\ \hline & 2 \leftarrow r \end{array}$$

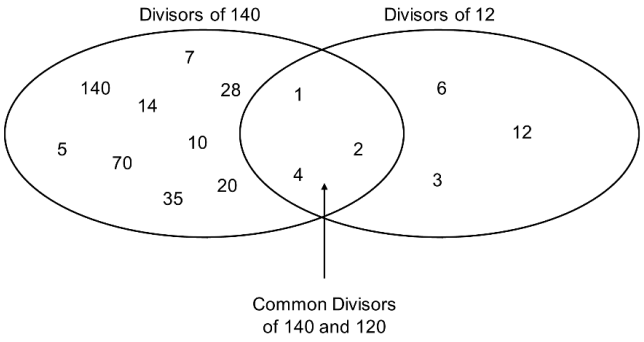


암호 수학

- 정수 연산(Integer Arithmetic) (3/12)
  - 가분성(Divisibility) (1/7)
    - 나눌 수 있는 성질
      - $a = q \times n$
      - $r = 0$
      - e.g., $32 = 8 \times 4$ 으로, $4 \mid 32$라고 표현
      - e.g., 나누어지지 않는 경우, $8 \nmid 42$
    - 성질
      - 만약 $a \mid 1$ 이면, $a = \pm 1$
      - 만약 $a \mid b$ 이고 $b \mid a$ 이면, $a = \pm b$
      - 만약 $a \mid b$ 이고 $b \mid c$ 이면, $a \mid c$
      - 만약 $a \mid b$ 이고 $a \mid c$ 이면, $a \mid (m \times b + n \times c)$
        - m과 n은 임의의 정수
      - e.g., $3 \mid 15$이고 $15 \mid 45$일 경우, 세 번째 성질에 의해 $3 \mid 45$
      - e.g., $3 \mid 15$이고 $3 \mid 9$이기 때문에, 네 번째 성질에 의해 $3 \mid (15 \times 2 + 9 \times 4)$이고, 여기서 m과 n은 임의의 정수

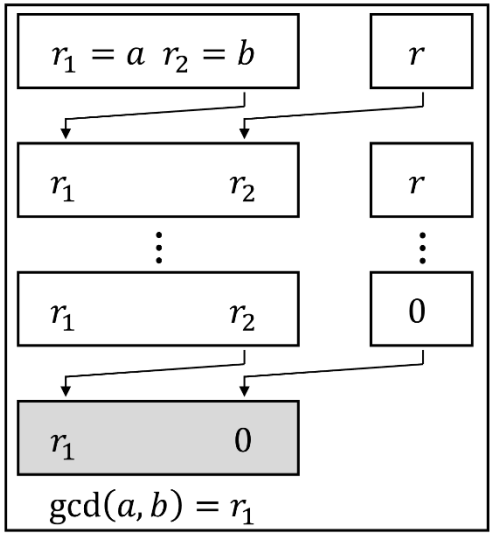
암호 수학

- 정수 연산(Integer Arithmetic) (4/12)
  - 가분성(Divisibility) (2/7)
    - 모든 약수
      - 양의 정수는 하나 이상의 약수(Divisor)를 가짐
        - e.g., 정수 32의 경우, 1, 2, 4, 8, 16, 32인 6개의 약수를 가짐
      - 정수 1은 하나의 약수, 1만을 가짐
      - 모든 양의 정수는 최소 2개의 약수를 가짐
        - 1과 자기 자신
    - 최대공약수(GCD, Greatest Common Divisor)
      - 두 양의 정수의 공통된 약수 중 가장 큰 수
        - 오직 하나의 최대공약수를 가짐
        - e.g., 12와 140의 공약수는 1, 2, 4이지만 최대공약수는 4 $(\gcd(140, 12) = 4)$



암호 수학

- 정수 연산(Integer Arithmetic) (5/12)
  - 가분성(Divisibility) (3/7)
    - 유클리드 알고리즘(Euclidean Algorithm)
      - 두 양의 정수의 최대공약수를 찾는 알고리즘
      - 두 사실에 기반
        - $\gcd(a, 0) = a$
        - $\gcd(a, b) = \gcd(b, r)$, 이때 r은 a를 b로 나눈 나머지
        - e.g., $\gcd(36, 10) = \gcd(10, 6) = \gcd(6, 4) = \gcd(4, 2) = \gcd(2, 0) = 2$



Process

```
Euclidean Algorithm() {
    r1 ← a;  r2 ← b;
    while (r2 > 0)
    {
        q ← r1/r2 ;
        r ← r1 – q × r2;
        r1 ← r2;  r2 ← r;
    }
    gcd(a, b) ← r1
}
```

- 정수 연산(Integer Arithmetic) (6/12)
  - 가분성(Divisibility) (4/7)
    - 유클리드 알고리즘(Euclidean Algorithm)
      - 예제 2.1

최대공약수를 구하시오.
1. $gcd(2470, 1760)$
2. $gcd(60, 25)$

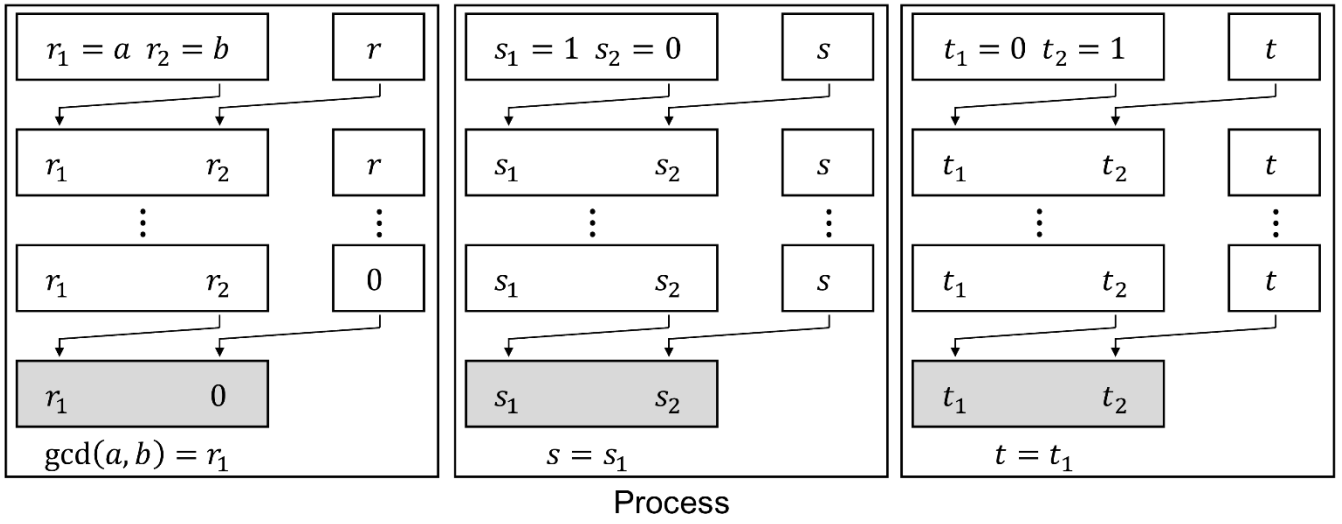
- 풀이
  - (1)

q	r_1	r_2	r
1	2740	1760	980
1	1760	980	780
1	980	780	200
3	780	200	180
1	200	180	20
9	180	20	0
	20	0	

- 풀이
  - (2)

q	r_1	r_2	r
0	25	60	25
2	60	25	10
2	25	10	5
2	10	5	0
	5	0	

- 정수 연산(Integer Arithmetic) (7/12)
  - 가분성(Divisibility) (5/7)
    - 확장 유클리드 알고리즘(Extended Euclidean Algorithm)
      - 두 정수 a, b가 주어질 때, 다음을 만족하는 다른 두 정수 s, t 계산
        - $s \times a + t \times b = \gcd(a, b)$
      - $\gcd(a, b)$와 s, t 동시에 계산 가능
      - 유클리드 알고리즘과 동일한 개수의 단계 사용하지만, 각 단계에서 한 쌍 대신 세 쌍의 계산과 교환 사용



Extended Euclidean Algorithm() {
  $r_1 \leftarrow a;\ r_2 \leftarrow b;$
  $s_1 \leftarrow 1;\ s_2 \leftarrow 0;$
  $t_1 \leftarrow 0;\ t_2 \leftarrow 1;$
  while $(r_2 > 0)$
  {
    $q \leftarrow r_1 / r_2\ ;$
    $r \leftarrow r_1 - q \times r_2;$
    $r_1 \leftarrow r_2;\ r_2 \leftarrow r;$
    $s \leftarrow s_1 - q \times s_2;$
    $s_1 \leftarrow s_2;\ s_2 \leftarrow s;$
    $t \leftarrow t_1 - q \times t_2;$
    $t_1 \leftarrow t_2;\ t_2 \leftarrow t;$
  }
  $\gcd(a, b) \leftarrow r_1;\ s \leftarrow s_1;\ t \leftarrow t_1$
}

- 정수 연산(Integer Arithmetic) (8/12)
  - 가분성(Divisibility) (6/7)
    - 확장 유클리드 알고리즘(Extended Euclidean Algorithm)
      - 예제 2.2 (1/2)

다음 조건일 경우, s, t를 구하시오.

1. $\mathrm{a} = 161, b = 28$
2. $\mathrm{a} = 17, b = 0$
3. $\mathrm{a} = 0, b = 45$

- 풀이
  - (1)

q	r_1	r_2	r	s_1	s_2	s	t_1	t_2	t
5	161	28	1	0	1	1	0	1	-5
1	28	21	7	0	1	-1	1	-5	6
3	21	7	0	1	-1	4	-5	6	-23
	7	0		**-1**	4		**6**	-23	

- 정수 연산(Integer Arithmetic) (9/12)
  - 가분성(Divisibility) (7/7)
    - 확장 유클리드 알고리즘(Extended Euclidean Algorithm)
      - 예제 2.2 (2/2)

다음 조건일 경우, s, t를 구하시오.

1. $\mathrm{a} = 161, b = 28$
2. $\mathrm{a} = 17, b = 0$
3. $\mathrm{a} = 0, b = 45$

- 풀이
  - (2)

q	r_1	r_2	r	s_1	s_2	s	t_1	t_2	t
	17	0		**1**	0		**0**	1	

  - (3)

q	r_1	r_2	r	s_1	s_2	s	t_1	t_2	t
0	0	45	0	1	0	1	0	1	0
	45	0		**0**	1		**1**	0	

암호 수학

- 정수 연산(Integer Arithmetic) (10/12)
  - 선형 디오판투스 방정식(Liner Diophantine Equations) (1/3)
    - 정의
      - 두 변수를 가진 $ax + by = c$ 형태의 방정식
    - 특징
      - 확장 유클리드 알고리즘 이용
      - $d = \gcd(a,\ b)$일 경우, d와 c를 통해 해 존재 여부 판별
        - $d \nmid c$일 경우, 방정식의 해는 존재하지 않음
        - $d \mid c$일 경우, 무수히 많은 해가 존재하며, 해들 중 하나를 특수(Particular) 해, 나머지를 일반(General) 해라고 함

암호 수학

- 정수 연산(Integer Arithmetic) (11/12)
  - 선형 디오판투스 방정식(Liner Diophantine Equations) (2/3)
    - 해 계산
      - 특수 해
        1. d로 방정식 양변을 나누어, $a_1x + b_1y = c1$으로 만듦
        2. 확장 유클리드 알고리즘을 사용하여, $a_1s + b_1t = 1$을 만족하는 s와 t 계산
        3. 특수 해는 $x_0 = (c/d)s,\ y_0 = (c/d)t$로 계산 가능
      - 일반 해
        - 특수 해를 찾아낸 뒤, 일반 해는 다음과 같음
        - $x = x_0 + k(b/d),\ y = y_0 - k(b/d)$, 이때 k는 정수
    - 활용
      - 다른 값들을 갖는 물체의 여러 조합을 탐색할 경우 응용
        - 100달러 수표를 20달러와 5달러 짜리로 현금화 할 경우, $200x + 5y = 100$을 계산하여 조합 탐색
          - $d = \gcd(20, 5) = 5$이고 $5|100$이므로 무수히 많은 해 존재
          - 특수 해는 $x_0 = 020 = 0, y_0 = 120 = 20$
          - 일반 해는 $(0, 20), (1, 16), (2, 12), (3, 8), (4, 4), (5, 0)$

- 정수 연산(Integer Arithmetic) (12/12)
  - 선형 디오판투스 방정식(Liner Diophantine Equations) (3/3)
    - 예제 2.3

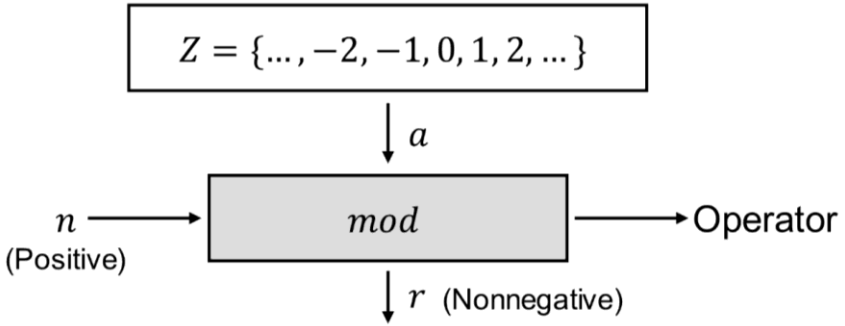
방정식 $21x + 14y = 35$의 특수 해와 일반 해를 구하시오.

- 풀이
  - $d = \gcd(21, 14) = 7$
  - $7 \mid 35$이므로 방정식은 무수히 많은 해를 가짐
  - 양변을 7로 나누면 $3x + 2y = 5$
  - 확장 유클리드 알고리즘을 사용하여 $3s + 2t = 1$를 만족하는 s, t를 계산
    - $s = 1, t, = -1$

∴
특수 해: $x_0 = 5\times1 = 5, y_0 = 5\times(-1) = -5$ ($35/7 = 5$ 이기 때문)
일반 해: $x = 5 + k\times2, y = -5 - k\times3$ (k는 정수)

암호 수학

- 모듈로 연산(Modular Arithmetic) (1/18)
  - 개요
    - $a = q \times n + r$에서 나머지에만 집중한 연산
  - 모듈로 연산자(Modulo Operator) (1/2)
    - mod로 표기
      - e.g., $12\ mod\ 5 = 2$
    - mod는 정수 집합 Z에서 a를 선택하고 양의 정수 모듈로(n)을 선택하여 음이 아닌 정수 r을 생성
      - 음인 r이 나올 경우, 모듈로(n)을 더하여 음이 아닌 정수 생성



암호 수학

- 모듈로 연산(Modular Arithmetic) (2/18)
  - 모듈로 연산자(Modulo Operator) (2/2)
    - 예제 2.4

다음을 계산하시오.

1. $27\ mod\ 5$
2. $36\ mod\ 12$
3. $-18\ mod\ 14$
4. $-7\ mod\ 10$

- 풀이
  (1) 27을 5로 나누면 나머지는 2이므로, $27\ mod\ 5 = 2$
  (2) 36을 12로 나누면 나머지는 0이므로, $36\ mod\ 12 = 0$
  (3) -18을 14로 나누면 나머지는 -4
    - 모듈로(14)를 더하여 음이 아닌 수로 만들어야 하므로, $-4 + 14 = 10$이므로 $-18\ mod\ 14 = 10$

  (4) -7을 10으로 나누면 나머지는 -7
    - 모듈로(10)을 더하여 음이 아닌 수로 만들어야 하므로, $-7 + 10 = 3$이므로 $-7\ mod\ 10 = 3$

암호 수학

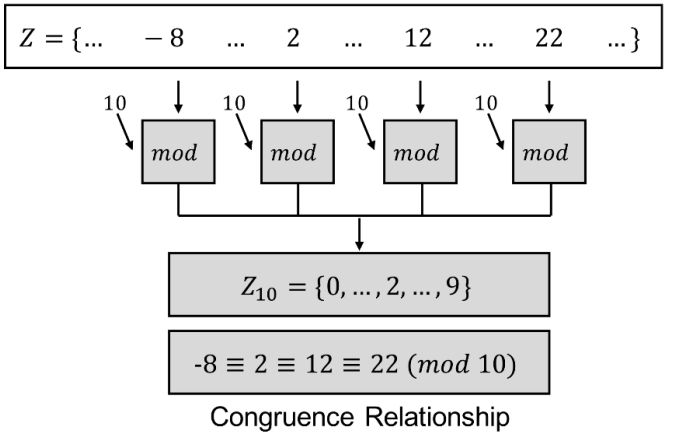
- 모듈로 연산(Modular Arithmetic) (3/18)
  - 잉여류(Residue Class) $[a]$

*최소 잉여 집합(Set of Least Residue Modulon) Z_n
모듈로 연산을 통해 나온 나머지들의 집합

    - 정의
      - n으로 나눈 나머지가 같은 정수들의 집합
        - $[a] = \{x : x \equiv a (mod\ n)\}$
    - 특징
      - 모듈로 n는 $[0], [1], \dots, [n-1]$의 n개의 잉여류 존재
      - 각 잉여류는 무한히 많은 정수 포함
      - 각 잉여류의 가장 작은 음이 아닌 정수가 그 잉여류를 대표하며, 대푯값들을 모은 집합이 *최소 잉여 집합 Z_n이라고 표현
    - 예시
      - $n = 3$일 경우
        - $[0] = \{\dots, -6, -3, 0, 3, 6, \dots\}, [1] = \{\dots, -5, -2, 1, 4, 7, \dots\}, [2] = \{\dots, -4, -1, 2, 5, 8, \dots\}$

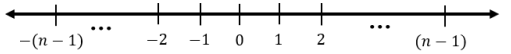
암호 수학

- 모듈로 연산(Modular Arithmetic) (4/18)
  - 합동(Congruence) (1/2)
    - 정의
      - 모듈로 n 연산 시, 나머지 같은 수들에 대한 관계
    - 특징
      - 합동 연산자(Congruence Operator) ≡ 사용
      - Z의 무한한 원소는 Z_n의 한 원소로 대응 가능
        - e.g., $2\ mod\ 10 = 2,\ 12\ mod\ 10 = 2,\ 22\ mod\ 10 = 2$일 때, 2, 12, 22은 $mod\ 10$에 대해 합동이므로 $2 \equiv 12 \equiv 22\ (mod\ 10)$로 표기 가능
    - 합동 연산자와 등식 연산자의 차이점
      - 등식 연산자
        - Z의 원소를 Z의 원소로 대응(Mapping)
        - 일-대-일(One-to-One)
      - 합동 연산자
        - Z의 원소에서 Z_n의 원소로 대응
        - 다-대-일(Many-to-One)

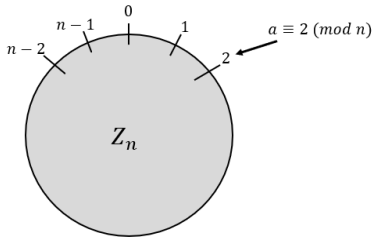


암호 수학

- 모듈로 연산(Modular Arithmetic) (5/18)
  - 합동(Congruence) (2/2)
    - 합동 연산자의 우변에 있는 $(mod\ n)$은 Z_n을 의미
      - 2진 연산과의 차이점
        - 2진 연산$(12\ mod\ 10)$에서 mod는 연산자
        - 합동$(2 \equiv 12(\mathrm{mod}10))$에서 $mod\ 10$은 공역 Z_{10}을 의미
    - 잉여류
      - 잉여류 $[a]$ 또는 $[a]n$는 모듈로 n으로 합동인 정수의 집합
      - $x = a\ (mod\ n)$을 만족하는 모든 정수의 집합
      - e.g., $n = 5$일 경우
        - $[5] = \{\dots, -15, -10, -5, 0, 5, 10, 15, \dots\}, [4] = \{\dots -14, -9, -4, 1, 6, 11, 16, \dots\},$ $[2] = \{\dots, -13, -8, -3, 2, 7, 12, 17, \dots\}, [1] = \{\dots, -12, -7, -2, 3, 8, 13, 18, \dots\},$ $[0] = \{\dots, -11, -6, -1, 4, 9, 14, 19, \dots\}$

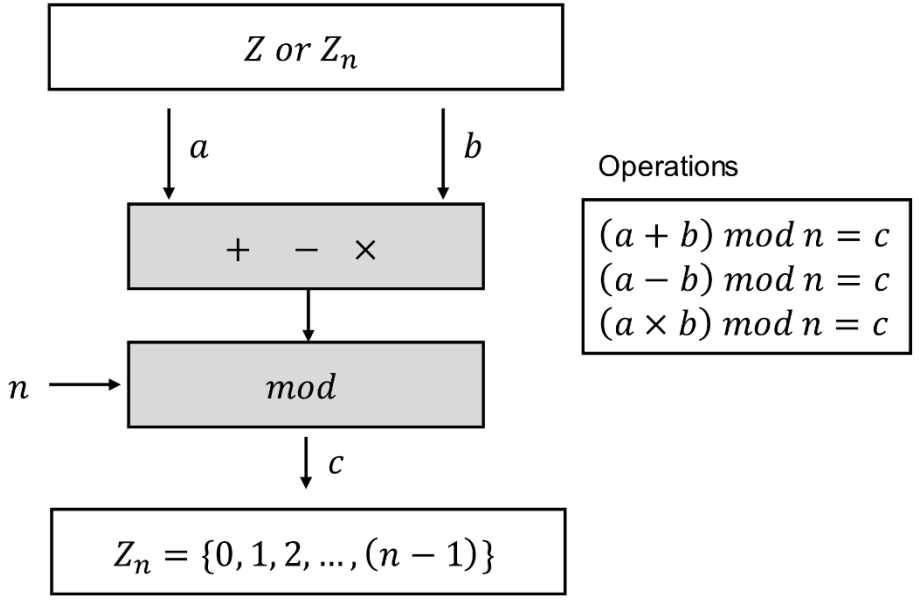


    - 원 기호
      - 0에서 $n-1$까지의 정수들은 원 주위에 고르게 분포되어 있으며, 양수와 음수는 대칭적으로 원에 대응



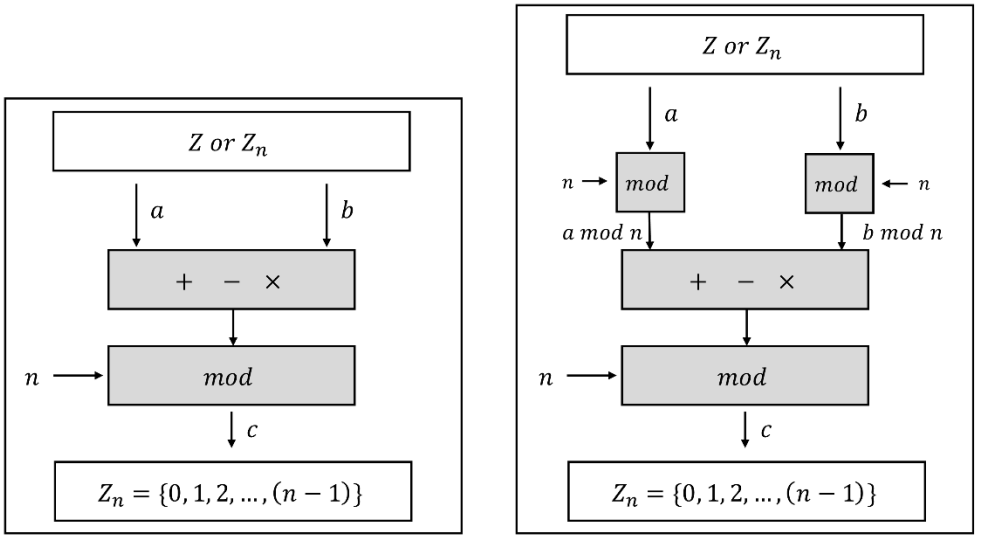
암호 수학

- 모듈로 연산(Modular Arithmetic) (6/18)
  - Z_n에서의 연산 (1/3)
    - Z_n에서 세 가지의 2진 연산(덧셈, 뺄셈, 곱셈) 정의
    - 연산자의 두 집합을 사용
      - 첫 번째 집합은 2진 연산자$(+, -, \times)$ 중 하나를 사용하는 것
      - 두 번째 집합은 mod 연산자를 사용하는 것
    - 연산자의 순서를 강조하기 위해 괄호 사용



암호 수학

- 모듈로 연산(Modular Arithmetic) (7/18)
  - Z_n에서의 연산 (2/3)
    - 성질
      - 세 개의 2진 연산($+, -, \times$)을 적용하기 전에 Z에서 선택될 경우, 두 입력 값을 Z_n에 우선 대응
      - $(a + b)\ mod\ n = [(a\ mod\ n) + (b\ mod\ n)]\ mod\ n$
      - $(a - b)\ mod\ n = [(a\ mod\ n) - (b\ mod\ n)]\ mod\ n$
      - $(a \times b)\ mod\ n = [(a\ mod\ n) \times (b\ mod\ n)]\ mod\ n$



- 모듈로 연산(Modular Arithmetic) (8/18)
  - Z_n에서의 연산 (3/3)
    - 예제 2.5

다음을 계산하시오.

1. Z_{15}의 7과 14를 더하시오.
2. Z_{13}의 7에서 11을 빼시오.
3. Z_{20}의 7과 11을 곱하시오.
4. Z_{14}의 17과 27를 더하시오.
5. Z_{13}의 12에서 43을 빼시오.
6. Z_{19}의 123과 -10을 곱하시오.

- 풀이

(1) $(14 + 7)\ mod\ 15 \rightarrow\ (21)\ mod\ 15 = 6$
(2) $(7 - 11)\ mod\ 13 \rightarrow\ (-4)\ mod\ 13 = 9$
(3) $(7{\times}11)\ mod\ 20 \rightarrow\ (77)\ mod\ 20 = 17$

(4) $(17 + 27)\ mod\ 14 \rightarrow\ (44)\ mod\ 14 = 2$
(5) $(12 - 43)\ mod\ 13 \rightarrow\ (-31)\ mod\ 13 = 8$
(6) $\big(123{\times}(-10)\big)\ mod\ 19 \rightarrow\ (-1230)\ mod\ 19\ =\ 5$

암호 수학

- 모듈로 연산(Modular Arithmetic) (9/18)
  - 역원(Inverse) (1/7)

*항등원(Identify)
어떤 원소에 특정 연산을 했을 경우, 자기 자신이 나오게 만드는 수

    - 정의
      - 특정 연산을 수행했을 때, 그 연산의 *항등원이 나오게 하는 원소
    - 종류
      - 덧셈에 대한 역원(Additive Inverse)
        - $a + b \equiv 0 \, mod \, n \; (\mathrm{Z_n}, \mathrm{a = n - b})$
        - Z_n에서 두 수 a, b는 서로 덧셈에 대한 역원
      - 곱셈에 대한 역원(Multiplicative Inverse)
        - $a \times b \equiv 1 \, mod \, n$
        - Z_n에서 두 수 a, b는 서로 곱셈에 대한 역원
        - a가 Z_n에서 곱셈에 대한 역원을 갖기 위한 필요충분조건은 $\gcd(n, a) = 1$이며, 이 경우 a와 n은 서로소

- 모듈로 연산(Modular Arithmetic) (10/18)
  - 역원(Inverse) (2/7)
    - 예제 2.6

Z_{10}에서 모든 덧셈에 대한 역원 쌍들을 찾으시오.

- 풀이
  - $1 = 10 - 9$
  - $2 = 10 - 8$
  - $3 = 10 - 7$
  - $4 = 10 - 6$
  - $5 = 10 - 5$
  - 0은 자신이 덧셈에 대한 역원

$\therefore (0,0), (1,9), (2,8), (3,7), (4,6), (5,5)$

암호 수학

- 모듈로 연산(Modular Arithmetic) (11/18)
  - 역원(Inverse) (3/7)
    - 예제 2.7

Z_{10}에서 8의 곱셈에 대한 역원을 계산하시오.

- 풀이
  - $\gcd(10, 8) = 2 \neq 1$이므로 곱셈에 대한 역원이 존재하지 않음

  $\therefore$ 존재하지 않음

    - 예제 2.8

Z_{10}에서 곱셈에 대한 모든 역원을 구하시오.

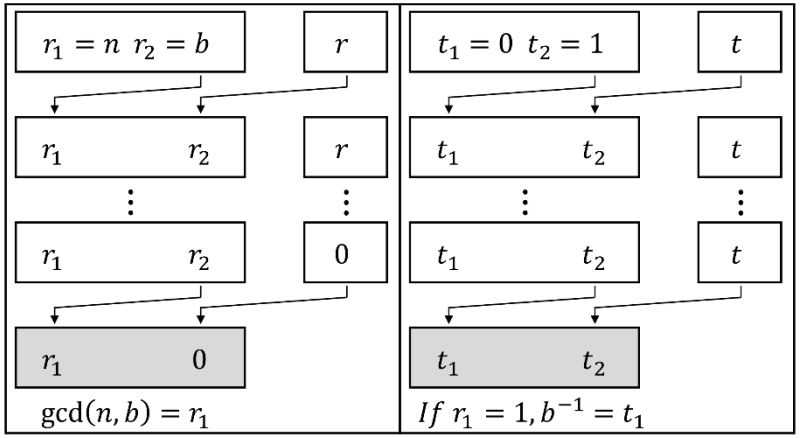
- 풀이
  - Z_{10}에서 곱셈에 대한 역원
    - $(1\times1)\ mod\ 10 = 1,\ (3\times7)\ mod\ 10 = 1,\ (9\times9)\ mod\ 10 = 1$
    - $0, 2, 4, 5, 6, 8$은 곱셈에 대한 역원이 존재하지 않음

  $\therefore (1, 1), (3, 7), (9, 9)$

암호 수학

- 모듈로 연산(Modular Arithmetic) (12/18)
  - 역원(Inverse) (4/7)
    - *확장 유클리드 알고리즘을 통한 곱셈에 대한 역원
      - n과 b가 주어지고 $\gcd(n,b)=1$일 때, $s\times n+t\times b=1$을 만족하는 정수 s,t를 확장 유클리드 알고리즘으로 계산
        1. $s\times a+t\times b=\gcd(a,b)$에서 a를 n으로 치환
           - $s\times n+t\times b=\gcd(n,b)$
        2. 역원이 존재할 경우, $\gcd(n,b)=1$이므로 $sn+tb=1$
        3. 양변에 모듈로 n 연산을 적용하면 $tb\equiv 1\ (mod\ n)$
        4. $t\ mod\ n$이 Z_n에서 b의 곱셈에 대한 역원

*확장 유클리드 알고리즘
두 정수 a,b가 주어질 때, 다음을 만족하는 다른 두 정수 s,t 계산.
$(s\times a+t\times b=\gcd(a,b))$



Process

Extended Euclidean Algorithm() {
  $r_1 \leftarrow a;\ r_2 \leftarrow b;$
  $t_1 \leftarrow 0;\ t_2 \leftarrow 1;$
  while ($r_2 > 0$)
  {
    $q \leftarrow r_1/r_2;$
    $r \leftarrow r_1 - q \times r_2;$
    $r_1 \leftarrow r_2;\ r_2 \leftarrow r;$
    $t \leftarrow t_1 - q \times t_2;$
    $t_1 \leftarrow t_2;\ t_2 \leftarrow t;$
  }
  $If\ (r_1 = 1)\ then\ b^{-1} \leftarrow t_1$
}

암호 수학

- 모듈로 연산(Modular Arithmetic) (13/18)
  - 역원(Inverse) (5/7)
    - 예제 2.9

Z_{26}에서 11의 곱셈에 대한 역원을 구하시오.

- 풀이
  - 확장 유클리드 알고리즘 사용 $(s \times a + t \times b = \gcd(a, b))$
  - $r_1 = n = 26, r_2 = b = 11$
  - $r_2 > 0$일 때까지 아래 반복
    - $q \leftarrow r_1 / r_2$
    - $r \leftarrow r_1 - q \times r_2$
    - $r_1 \leftarrow r_2,\ r_2 \leftarrow r$
    - $t \leftarrow t_1 - q \times t_2$
    - $t_1 \leftarrow t_2,\ t_2 \leftarrow t$
  - 만약 $r_1 = 1$이 될 경우, $t_1 = -6$
    - $(-7) mod\ 26 = 19$

  $\therefore 11, 19$

q	r_1	r_2	r	t_1	t_2	t
2	26	11	4	0	1	-2
2	11	4	3	1	-2	5
1	4	3	1	-2	5	-7
3	3	1	0	5	-7	26
	1	0		-7	26	

암호 수학

- 모듈로 연산(Modular Arithmetic) (14/18)
  - 역원(Inverse) (6/7)
    - 예제 2.10

Z_{100}에서 23의 곱셈에 대한 역원을 계산하시오.

- 풀이
  - 확장 유클리드 알고리즘 사용 $(s \times a + t \times b = \gcd(a, b))$
  - $r_1 = n = 100, r_2 = b = 23$
  - $r_2 > 0$일 때까지 아래 반복
    - $q \leftarrow r_1 / r_2$
    - $r \leftarrow r_1 - q \times r_2$
    - $r_1 \leftarrow r_2,\ r_2 \leftarrow r$
    - $t \leftarrow t_1 - q \times t_2$
    - $t_1 \leftarrow t_2,\ t_2 \leftarrow t$
  - 만약 $r_1 = 1$이 될 경우, $t_1 = -13$
    - $(-13) mod\ 100 = 87$

$\therefore 23, 87$

q	r_1	r_2	r	t_1	t_2	t
4	100	23	8	0	1	-4
2	23	8	7	1	-4	19
1	8	7	1	-4	9	-13
7	7	1	0	9	-13	100
	1	0		**-13**	100	

- 모듈로 연산(Modular Arithmetic) (15/18)
  - 역원(Inverse) (7/7)
    - 예제 2.11

Z_{26}에서 12의 곱셈에 대한 역원을 계산하시오.

- 풀이
  - 확장 유클리드 알고리즘 사용 $(s \times a + t \times b = \gcd(a, b))$
  - $r_1 = n = 26, r_2 = b = 12$
  - $r_2 > 0$일 때까지 아래 반복
    - $q \leftarrow r_1 / r_2$
    - $r \leftarrow r_1 - q \times r_2$
    - $r_1 \leftarrow r_2,\ r_2 \leftarrow r$
    - $t \leftarrow t_1 - q \times t_2$
    - $t_1 \leftarrow t_2,\ t_2 \leftarrow t$

q	r_1	r_2	r	t_1	t_2	t
2	26	12	2	0	1	-2
6	12	2	0	1	-2	13
	2	0		-2	13	

  - $\gcd(26, 12) = 2 \neq 1$이므로 곱셈에 대한 역원이 존재하지 않음

  $\therefore$ 곱셈에 대한 역원이 존재하지 않음

암호 수학

- 모듈로 연산(Modular Arithmetic) (16/18)
  - 덧셈표와 곱셈표(Addition and Multiplication Tables)
    - Z_n에서의 연산 결과를 표 형태로 나타낸 것
    - 덧셈표
      - 각각의 정수는 덧셈에 대한 역원을 가짐
      - 교환성
        - $a + b = b + a$
      - 각각의 행과 열이 다른 행과 열의 순열(Permutation)
    - 곱셈표
      - $(1,1), (3,7), (9,9)$의 경우만 역원 존재
        - 곱셈 결과가 1인 경우 역원 존재
      - 교환성
        - $a \times b = b \times a$

	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	0
2	2	3	4	5	6	7	8	9	0	1
3	3	4	5	6	7	8	9	0	1	2
4	4	5	6	7	8	9	0	1	2	3
5	5	6	7	8	9	0	1	2	3	4
6	6	7	8	9	0	1	2	3	4	5
7	7	8	9	0	1	2	3	4	5	6
8	8	9	0	1	2	3	4	5	6	7
9	9	0	1	2	3	4	5	6	7	8

	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	0	2	4	6	8
3	0	3	6	9	2	5	8	1	4	7
4	0	4	8	2	6	0	4	8	2	6
5	0	5	0	5	0	5	0	5	0	5
6	0	6	2	8	4	0	6	2	8	4
7	0	7	4	1	8	5	2	9	6	3
8	0	8	6	4	2	0	8	6	4	2
9	0	9	8	7	6	5	4	3	2	1

암호 수학

- 모듈로 연산(Modular Arithmetic) (17/18)
  - 덧셈과 곱셈에 대한 다른 집합
    - 개요
      - Z_n은 모듈로 n에서 가능한 모든 나머지의 집합
      - Z_n^*은 Z_n에서 곱셈에 대한 역원을 갖는 원소들의 집합
        - $Z_n^* = \{a \in Z_n \mid \gcd(a, n) = 1\}$
    - 덧셈과 곱셈의 차이
      - Z_n의 모든 원소는 덧셈에 대한 역원을 가지므로 덧셈 연산은 Z_n 전체에서 가능
      - 곱셈에 대한 역원은 $gcd(a, n) = 1$의 원소에만 존재하므로 Z_n^*는 Z_n의 부분집합
    - 예시
      - $Z_6 = \{0, 1, 2, 3, 4, 5\}$, $Z_6^* = \{1, 5\}$
      - $Z_7 = \{0, 1, 2, 3, 4, 5, 6\}$, $Z_7^* = \{1, 2, 3, 4, 5, 6\}$
      - $Z_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$, $Z_{10}^* = \{1, 3, 7, 9\}$

암호 수학

- 모듈로 연산(Modular Arithmetic) (18/18)
  - 모듈로가 소수(Prime Number)인 집합
    - 개요
      - p가 소수이면 $Z_p = \{0,1,2,\dots,p-1\}$으로 Z_n과 동일
        - $Z_p^* = \{1,2,\dots,p-1\}$
    - 역원의 존재
      - Z_p의 모든 원소는 덧셈에 대한 역원을 가짐
      - Z_p의 0이 아닌 모든 원소는 곱셈에 대한 역원을 가짐
    - 집합의 크기
      - $|Z_p| = p$
      - $|Z_p^*| = p-1$
    - 예시
      - $Z_7 = \{0,1,2,3,4,5,6\}$
      - $Z_7^* = \{1,2,3,4,5,6\}$

암호 수학

- 행렬(Matrices) (1/9)
  - 정의
    - $l \times m$개의 원소를 갖는 직사각형 배열
  - 특징
    - l은 행의 개수, m은 열의 개수
    - $\boldsymbol{A}$와 같은 볼드체의 대문자로 표기
    - a_{ij}는 i번째 행과 j번째 열에 위치한 원소

$$
\begin{array}{cc}
 & m \text{ columms} \\
l \text{ rows} & \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & & \vdots \\ a_{l1} & a_{l2} & \cdots & a_{lm} \end{bmatrix}
\end{array}
$$

$$Matrix\ \boldsymbol{A}$$

암호 수학

- 행렬(Matrices) (2/9)
  - 종류
    - 행 행렬(Row Matrix)
      - 하나의 행($l = 1$)만 갖는 행렬
    - 열 행렬(Column Matrix)
      - 하나의 열($m = 1$)만 갖는 행렬
    - 정방 행렬(Square Matrix)
      - 행의 개수와 열의 개수가 같은 행렬($l = m$)
    - 영 행렬(Zero Matrix)
      - 행렬의 모든 원소가 0인 행렬
    - 항등 행렬(Identity Matrix)
      - *주대각선이 1이고 나머지는 0인 정방 행렬

*주대각선(Main Diagonal)
정방행렬에서 원소 $a_{11}, a_{22}, a_{33} \ldots, a_{mm}$이 이루는 선

Row Matrix $\begin{bmatrix} 2 & 1 & 5 & 11 \end{bmatrix}$

Column Matrix $\begin{bmatrix} 2 \\ 4 \\ 12 \end{bmatrix}$

Square Matrix $\begin{bmatrix} 23 & 14 & 2 \\ 12 & 21 & 4 \\ 10 & 8 & 12 \end{bmatrix}$

Zero Matrix $\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$

Identity Matrix $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

암호 수학

- 행렬(Matrices) (3/9)
  - 관계식(Relation)
    - 등식(Equality)
      - 두 행렬의 행과 열의 개수가 대응되는 원소가 동일
        - 모든 i와 j에 대해 $a_{ij} = b_{ij}$일 경우, $\boldsymbol{A} = \boldsymbol{B}$로 표기
      - e.g., $[2 \quad 1] = [2 \quad 1]$
  - 연산(Operation) (1/2)
    - 덧셈과 뺄셈
      - 두 행렬의 행과 열의 개수가 동일할 경우, 덧셈/뺄셈 가능
      - $\boldsymbol{C} = \boldsymbol{A} \pm \boldsymbol{B}$로 표기
        - $c_{ij} = a_{ij} \pm b_{ij}$
      - 생성된 $\boldsymbol{C}$는 $\boldsymbol{A}, \boldsymbol{B}$와 같은 개수의 행과 열 소유
      - e.g., $\begin{bmatrix} 12 & 4 & 4 \\ 11 & 12 & 30 \end{bmatrix} = \begin{bmatrix} 5 & 2 & 1 \\ 3 & 2 & 10 \end{bmatrix} + \begin{bmatrix} 7 & 2 & 3 \\ 8 & 10 & 20 \end{bmatrix}$
      - e.g., $\begin{bmatrix} -2 & 0 & -2 \\ -5 & -8 & 10 \end{bmatrix} = \begin{bmatrix} 5 & 2 & 1 \\ 3 & 2 & 10 \end{bmatrix} - \begin{bmatrix} 7 & 2 & 3 \\ 8 & 10 & 20 \end{bmatrix}$

암호 수학

- 행렬(Matrices) (4/9)
  - 연산(Operation) (2/2)
    - 곱셈
      - 첫 번째 행렬의 열의 개수와 두 번째 행렬의 행의 개수가 같을 경우, 곱셈 가능
      - $\boldsymbol{A}$가 $l \times m$ 행렬이고, $\boldsymbol{B}$가 $m \times p$ 행렬일 경우, 두 행렬의 곱은 $l \times p$ 행렬
      - e.g., $\begin{bmatrix} 52 & 18 & 14 & 9 \\ 41 & 21 & 22 & 7 \end{bmatrix} = \begin{bmatrix} 5 & 2 & 1 \\ 3 & 2 & 4 \end{bmatrix} \times \begin{bmatrix} 7 & 3 & 2 & 1 \\ 8 & 0 & 0 & 2 \\ 1 & 3 & 4 & 0 \end{bmatrix}$
    - *스칼라 곱
      - 행렬에 스칼라를 곱하는 것
      - $\boldsymbol{A}$가 $l \times m$ 행렬이고 x가 스칼라일 경우, $\boldsymbol{C} = x\boldsymbol{A}$
        - $c_{ij} = x \times a_{ij}$인 $l \times m$ 행렬
      - e.g., $\begin{bmatrix} 15 & 6 & 3 \\ 9 & 6 & 12 \end{bmatrix} = 3 \times \begin{bmatrix} 5 & 2 & 1 \\ 3 & 2 & 4 \end{bmatrix}$

*스칼라(Scalar)
방향 없이 크기(단일 숫자 값)만으로 나타낼 수 있는 양

암호 수학

- 행렬(Matrices) (5/9)
  - 행렬식(Determinant)
    - 크기가 $m \times m$인 정방 행렬 $\boldsymbol{A}$의 행렬식은 $\det(\boldsymbol{A})$로 표기
      - $m = 1$인 경우, $\det(\boldsymbol{A}) = a_{11}$
      - $m > 1$인 경우, $\det(\boldsymbol{A}) = \sum_{i=1,\ldots,m}(-1)^{i+j} \times a_{ij} \times \det(\boldsymbol{A}_{ij})$
        - $\boldsymbol{A}_{ij}$는 $\boldsymbol{A}$에서 i열과 j행을 제거한 뒤 얻어진 행렬
      - e.g.,

$$\det\begin{bmatrix} 5 & 2 & 1 \\ 3 & 0 & -4 \\ 2 & 1 & 6 \end{bmatrix}$$

$$= (-1)^{1+1} \times 5 \times det\begin{bmatrix} 3 & -4 \\ 2 & 6 \end{bmatrix} + (-1)^{1+3} \times 1 \times det\begin{bmatrix} 3 & 0 \\ 2 & 1 \end{bmatrix}$$

$$= (+1) \times 5 \times (+4) + (-1) \times 2 \times (24) + (+1) \times 1 \times (3) = -25$$

암호 수학

- 행렬(Matrices) (6/9)
  - 역행렬(Inverse Matrix) (1/2)
    - 정의
      - 정방 행렬에 곱했을 때 항등 행렬이 나오게 하는 행렬
    - 특징
      - 정방 행렬 $\boldsymbol{A}$에서 $\det(\boldsymbol{A}) \neq 0$일 경우, 역행렬이 존재
      - 역행렬이 존재할 경우, 행렬 별로 하나만 존재
      - $\left(\boldsymbol{A}^{-1}\right)^{-1} = \boldsymbol{A}$
      - $(\boldsymbol{AB})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1}$
      - $(\boldsymbol{A}^{T})^{-1} = (\boldsymbol{A}^{-1})^{T}$
      - $\det(\boldsymbol{A}) = 0$일 경우, 역행렬은 존재하지 않음
    - 구분
      - 덧셈에 대한 역행렬
      - 곱셈에 대한 역행렬

암호 수학

- 행렬(Matrices) (7/9)
  - 역행렬(Inverse Matrix) (2/2)
    - 덧셈에 대한 역행렬
      - 행렬 $\boldsymbol{A}$의 덧셈에 대한 역행렬은 $\boldsymbol{A}+\boldsymbol{B}=0$을 만족하는 또 다른 행렬 $\boldsymbol{B}$
      - 모든 i와 j에 대해 $b_{ij}=-a_{ij}$이고 $\boldsymbol{A}$의 덧셈에 대한 역행렬은 $-\boldsymbol{A}$
      - e.g., $\boldsymbol{A}=\begin{bmatrix}2 & 5\\ 4 & 3\end{bmatrix}$, $\boldsymbol{B}=\begin{bmatrix}-2 & -5\\ -4 & -3\end{bmatrix}$ $\therefore \boldsymbol{B}=-\boldsymbol{A}$
    - 곱셈에 대한 역행렬
      - 정방 행렬 $\boldsymbol{A}$의 곱셈에 대한 역행렬은 $\boldsymbol{A}\times\boldsymbol{B}=\boldsymbol{B}\times\boldsymbol{A}=\boldsymbol{I}$를 만족하는 정방 행렬 $\boldsymbol{B}$로 $\boldsymbol{A}^{-1}$로 표기
      - $\det(\boldsymbol{A})$가 대응되는 집합에서 곱셈에 대한 역원을 가질 경우에만 존재
      - 어떤 정수도 Z에서 곱셈에 대한 역원을 갖지 않기 때문에, Z에서는 행렬의 곱셈에 대한 역행렬이 존재하지 않음
      - 실수를 원소로 하는 행렬은 $\det(\boldsymbol{A})\neq 0$일 경우에만 역행렬을 가짐
      - e.g., $\begin{bmatrix}2 & 5\\ 1 & 3\end{bmatrix}$의 역행렬은 $\begin{bmatrix}3 & -5\\ -1 & 2\end{bmatrix}$

암호 수학

- 행렬(Matrices) (8/9)
  - 잉여 행렬(Residue Matrices) (1/2)
    - 정의
      - 모든 행렬의 원소가 Z_n에 속하는 행렬
    - 특징
      - 행렬의 연산은 모듈로 n에 대한 연산으로 수행
      - 행렬식이 Z_n에서 곱셈에 대한 역원을 가질 경우, 곱셈에 대한 역행렬 존재
      - $\gcd(\det(\boldsymbol{A}), n) = 1$일 경우, 곱셈에 대한 역행렬 $\boldsymbol{A}^{-1}$ 존재
    - 예시
      - Z_6에서 다음 행렬 A가 주어질 경우
        - $\boldsymbol{A} = \begin{bmatrix} 1 & 2 \\ 3 & 5 \end{bmatrix},\ \det(\boldsymbol{A}) = 1\times5 - 2\times3 = -1,\ \gcd(\det(\boldsymbol{A}), 6) = 1$
        - $\boldsymbol{A}^{-1} = \begin{bmatrix} 5 & 4 \\ 3 & 5 \end{bmatrix}\ (mod\ 6)$

암호 수학

- 행렬(Matrices) (9/9)
  - 잉여 행렬(Residue Matrices) (2/2)
    - 합동
      - 두 행렬의 행과 열의 개수가 동일하며 대응하는 모든 원소가 모듈로 n에 대하여 합동인 관계
        - $\boldsymbol{A} \equiv \boldsymbol{B} \ (mod\ n)$
      - 모든 i와 j에 대해 $a_{ij} \equiv b_{ij} \ (mod\ n)$이면, $\boldsymbol{A} \equiv \boldsymbol{B} \ (mod\ n)$
      - 예시 – $mod\ 6$
        - $\boldsymbol{A} = \begin{bmatrix} 1 & 8 \\ 3 & 5 \end{bmatrix},\ \boldsymbol{B} = \begin{bmatrix} 7 & 2 \\ 9 & 11 \end{bmatrix}$
        - $1 \equiv 7,\ 8 \equiv 2,\ 3 \equiv 9,\ 5 \equiv 11$
        - $\boldsymbol{A} \equiv \boldsymbol{B} \ (mod\ 6)$

암호 수학

- 선형 합동 (1/6)
  - 일변수 선형 방정식(Single-variable Liner Equations) (1/2)
    - 변수가 하나인 방정식으로 $ax \equiv b \ (mod\ n)$ 형태의 방정식
      - 해가 없거나 유한개의 해가 존재함
    - $\gcd(a, n) = b$라고 가정
      - $d \nmid b$일 경우, 해가 존재하지 않음
      - $d \mid b$일 경우, 해가 존재함
    - 해가 존재할 경우 해를 구하는 방법
      1. 모듈로를 포함하여 방정식의 양변을 d로 나눔
      2. 특수 해 x_0를 구하기 위해 약분한 방정식의 양변에 a의 곱셈에 대한 역원을 곱함
      3. 일반 해는 $k = 0, 1, \ldots, (d-1)$에 대하여, $x = x_0 + k(n/d)$ 임

- 선형 합동 (2/6)
  - 일변수 선형 방정식(Single-variable Liner Equations) (2/2)
    - 예제 2.12

다음을 계산하시오.

1. 방정식 $10x \equiv 2\ (mod\ 15)$의 해를 구하시오.
2. 방정식 $14x \equiv 12\ (mod\ 18)$의 해를 구하시오.
3. 방정식 $3x + 4 \equiv 6\ (mod\ 13)$의 해를 구하시오.

- 풀이
  1. $\gcd(10, 15) = 5$이고 5가 2를 나누지 않기 때문에 해가 없음
  2. $\gcd(14, 18) = 2$이고 $2 \mid 12$이므로 2개의 해 존재
     - $14x \equiv 12\ (mod\ 18) \rightarrow 7x \equiv 6\ (mod\ 9) \rightarrow x \equiv 6(7^{-1})(mod\ 9)$
     - $x_0 = (6{\times}7^{-1})\ mod\ 9 = (6{\times}4)\ (mod\ 9) = 6$
     - $x_1 = x_0 + 1{\times}\left(\frac{18}{2}\right) = 15$
  3. $3x \equiv 2\ (mod\ 13)$으로 정리 후, $\gcd(3, 13) = 1$이고 오직 한 개의 근 존재
     - $x_0 = (2{\times}3^{-1}) mod\ 13 = 18\ mod\ 13 = 5$

암호 수학

- 선형 합동 (3/6)
  - 일차 연립 방정식 (1/4)
    - 동일한 모듈로 n에 대한 여러 개의 *선형 합동식으로 이루어진 방정식 체계

*선형 합동식
미지수의 일차식에 대한 합동식

    - 행렬의 곱으로 표현
      - 변수의 계수로 구성된 정방 행렬
      - 변수로 구성된 열 행렬
      - 등식의 우변 값으로 구성된 열 행렬

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &\equiv b_1 \\ a_{21}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &\equiv b_2 \\ \vdots \quad\quad \vdots \quad\quad\quad\quad \vdots \quad\quad \vdots & \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &\equiv b_n \end{aligned}$$

Equations

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \equiv \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Interpretation

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \equiv \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}^{-1} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Solution

암호 수학

- 선형 합동 (4/6)
  - 일차 연립 방정식 (2/4)
    - 변수의 계수로 구성된 정방 행렬의 역행렬이 존재할 경우, 공통 해 계산 가능함
      1. 연립 방정식을 계수만 있는 행렬과 변수만 있는 행렬로 분리함
      2. 계수만 있는 행렬의 역행렬 계산함
      3. 역행렬을 양변에 곱함
      4. 결과를 모듈로 연산 진행함

$$\boxed{\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \equiv b_1 \\ a_{21}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \equiv b_2 \\ \quad\vdots \qquad\quad \vdots \qquad\qquad\quad \vdots \qquad\quad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n \equiv b_n \end{array}}$$

Equations

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \equiv \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Interpretation

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \equiv \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}^{-1} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Solution

- 선형 합동 (5/6)
  - 일차 연립 방정식 (3/4)
    - 예제 2.13 (1/2)

다음 일차 연립 방정식의 해를 계산하시오.

$$5x + 5y + 7z \equiv 3 \ (mod\ 16)$$
$$x + 4y + 13z \equiv 5 \ (mod\ 16)$$
$$2x + 7y + 3z \equiv 4 \ (mod\ 16)$$

- 풀이
  - 연립 방정식을 행렬 $\boldsymbol{A}$로 바꿈
    - $\boldsymbol{A} = \begin{bmatrix} 3 & 5 & 7 \\ 1 & 4 & 13 \\ 2 & 7 & 3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 3 \\ 5 \\ 4 \end{bmatrix}$
  - 방정식의 집합으로 구성된 행렬은 역행렬이 존재하므로, 그 행렬의 곱셈에 대한 역행렬을 구하여 3, 5, 4로 구성된 열 행렬에 곱함
    - $\boldsymbol{A}^{-1} = \frac{1}{129} \begin{bmatrix} 79 & -34 & -37 \\ -23 & 5 & 32 \\ 1 & 11 & -7 \end{bmatrix}$, $\boldsymbol{A}\boldsymbol{A}^{-1} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \boldsymbol{A}^{-1} \begin{bmatrix} 3 \\ 5 \\ 4 \end{bmatrix} = \frac{1}{129} \begin{bmatrix} -81 \\ 84 \\ 30 \end{bmatrix}$

- 선형 합동 (6/6)
  - 일차 연립 방정식 (4/4)
    - 예제 2.13 (2/2)

다음 일차 연립 방정식의 해를 계산하시오.

$$5x + 5y + 7z \equiv 3 \ (mod\ 16)$$
$$x + 4y + 13z \equiv 5 \ (mod\ 16)$$
$$2x + 7y + 3z \equiv 4 \ (mod\ 16)$$

- 풀이
  - 이에 모듈로 연산 진행
    - $\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 15 \ (mod\ 16) \\ 4 \ (mod\ 16) \\ 14 \ (mod\ 16) \end{bmatrix}$

$\therefore x \equiv 15 \ (mod\ 16),\ y \equiv 4 \ (mod\ 16),\ z \equiv 14 \ (mod\ 16)$

Thanks!

서 진 우(jinwoo@pel.sejong.ac.kr)

부록#1

- 공격 비교
  - 스누핑(Snooping)
    - 비인가 접근을 통해 데이터를 무단으로 열람하거나 탈취하는 공격
    - e.g., 다른 사람의 계정에 무단으로 접속해 저장된 파일을 열람
  - 스푸핑(Spooffing)
    - 다른 사용자나 시스템으로 위장하여 상대를 속이는 공격
    - e.g., 은행을 사칭한 이메일을 보내 가짜 로그인 페이지에 비밀번호를 입력하도록 유도
  - 스니핑(Sniffing)
    - 통신 경로를 지나는 데이터를 캡처하여 내용을 엿보는 공격
    - e.g., 보안이 적용되지 않은 공용 와이파이에서 전송되는 로그인 정보를 캡처