E-MAIL SECURITY

Leonardo Querzoni
querzoni@diag.uniroma1.it

SAPIENZA
UNIVERSITÀ DI ROMA

CIS SAPIENZA
CYBER INTELLIGENCE AND INFORMATION SECURITY

SECURITY CHALLENGES

The email system suffers from several well-known threats:

- Spam
- Phishing Attacks
- Malware/Ransomware Distribution
- Email Spoofing
- Lack of traceability
- Data Leakage
- Man-in-the-Middle (MITM) Attacks
- Business Email Compromise (BEC)
- Email Bombing

SIMPLE SPOOFING EXAMPLE

```
Delivered-To: leonardo.querzoni@uniroma1.it
Received: by 2002:a05:6520:2e05:b0:2a1:68a6:a6c0 with SMTP id df5csp164156lkb;
        Tue, 17 Sep 2024 04:50:45 -0700 (PDT)
Return-Path: <alessandro.lazzaro@uniroma1.it>
Received: from mail-sor-f41.google.com (mail-sor-f41.google.com. [209.85.220.41])
        by mx.google.com with SMTPS id2adb3069b0e04-5368707b163sor1622092e87.11.2024.09.17.04.50.42
        for <leonardo.querzoni@uniroma1.it> (Google Transport Security); Tue, 17 Sep
        2024 04:50:42 -0700 (PDT)
In-Reply-To: <CAMSOQ=mLT1Vx5H5h8oVQnMz4nzFEkQ=UeMS0boAhf3v02tQLyw@mail.gmail.com>
Reply-To: servicecepol@gmail.com
From: Polizia criminale <alessandro.lazzaro@uniroma1.it>
Date: Tue, 17 Sep 2024 13:50:27 +0200
Message-ID: <CAMSOQ=kfzPNKa7gF3MAD_7hEKauRxxZVitjUSGC3nwsAkoGOGQ@mail.gmail.com>
Subject: ⭕RICHIESTA DI PROTEZIONE PERSONALE⭕
To: polizia-cr@info.it
Bcc: leonardo.querzoni@uniroma1.it
Content-Type: multipart/mixed; boundary="0000000000070a7ab06224f4cad"

--0000000000070a7ab06224f4cad
```

SIMPLE SPOOFING EXAMPLE

Delivered-To: leonardo.querzoni@uniroma1.it
Received: by 2002:a05:6520:2e05:b0:2a1:68a6:a6c0 with SMTP id df5csp164156lkb;
Tue, 17 Sep 2024 04:50:45 -0700 (PDT)
Return-Path: <alessandro.lazzaro@uniroma1.it>
Received: from mail-sor-f41.google.com (mail-sor-f41.google.com. [209.85.220.41])
by mx.google.com with SMTPS id2adb3069b0e04-5368707b163sor1622092e87.11.2024.09.17.04.50.42
for <leonardo.querzoni@uniroma1.it> (Google Transport Security); Tue, 17 Sep
2024 04:50:42 -0700 (PDT)
In-Reply-To: <CAMSOQ=mLT1Vx5H5h8oVQnMz4nzFEkQ=UeMS0boAhf3v02tQLyw@mail.gmail.com>
Reply-To: servicecepol@gmail.com
From: Polizia criminale <alessandro.lazzaro@uniroma1.it>
Date: Tue, 17 Sep 2024 13:50:27 +0200
Message-ID: <CAMSOQ=kfzPNKa7gF3MAD_7hEKauRxxZVitjUSGC3nwsAkoGOGQ@mail.gmail.com>
Subject: ⭕RICHIESTA DI PROTEZIONE PERSONALE⭕
To: polizia-cr@info.it
Bcc: leonardo.querzoni@uniroma1.it
Content-Type: multipart/mixed; boundary="0000000000070a7ab06224f4cad"

--0000000000070a7ab06224f4cad

SIMPLE SPOOFING EXAMPLE

Delivered-To: leonardo.querzoni@uniroma1.it
Received: by 2002:a05:6520:2e05:b0:2a1:68a6:a6c0 with SMTP id df5csp164156lkb;
Tue, 17 Sep 2024 04:50:45 -0700 (PDT)
Return-Path: <alessandro.lazzaro@uniroma1.it>
Received: from mail-sor-f41.google.com (mail-sor-f41.google.com. [209.85.220.41])
by mx.google.com with SMTPS id2adb3069b0e04-5368707b163sor1622092e87.11.2024.09.17.04.50.42
for <leonardo.querzoni@uniroma1.it> (Google Transport Security); Tue, 17 Sep
2024 04:50:42 -0700 (PDT)
In-Reply-To: <CAMSOQ=mLT1Vx5H5h8oVQnMz4nzFEkQ=UeMS0boAhf3v02tQLyw@mail.gmail.com>
Reply-To: servicecepol@gmail.com
From: Polizia criminale <alessandro.lazzaro@uniroma1.it>
Date: Tue, 17 Sep 2024 13:50:27 +0200
Message-ID: <CAMSOQ=kfzPNKa7gF3MAD_7hEKauRxxZVitjUSGC3nwsAkoGOGQ@mail.gmail.com>
Subject: ⭕RICHIESTA DI PROTEZIONE PERSONALE⭕
To: polizia-cr@info.it
Bcc: leonardo.querzoni@uniroma1.it
Content-Type: multipart/mixed; boundary="0000000000070a7ab06224f4cad"

--0000000000070a7ab06224f4cad

SIMPLE SPOOFING EXAMPLE 2

```
Delivered-To: querzoni@dis.uniroma1.it
Received: by 2002:a59:c26b:0:b0:48e:d422:933f with SMTP id c11csp2768947vqr;
        Tue, 24 Sep 2024 00:18:05 -0700 (PDT)
Return-Path: <leonardo.querzoni+caf_=querzoni=dis.uniroma1.it@uniroma1.it>
Received: from mail-sor-f41.google.com (mail-sor-f41.google.com. [209.85.220.41])
        by mx.google.com with SMTPS id 2adb3069b0e04-537a86516a3sor234209e87.25.2024.09.24.00.18.04
        for <querzoni@dis.uniroma1.it> (Google Transport Security);
        Tue, 24 Sep 2024 00:18:05 -0700 (PDT)
Return-Path: <gaia.fiore@uniroma1.it>
Received: from mail-sor-f41.google.com (mail-sor-f41.google.com. [209.85.220.41])
        by mx.google.com with SMTPS id 41be03b00d2f7-7e6b7c4586fsor419004a12.4.2024.09.24.00.18.02
        for <leonardo.querzoni@uniroma1.it> (Google Transport Security);
        Tue, 24 Sep 2024 00:18:03 -0700 (PDT)
MIME-Version: 1.0
From: "*SAPIENZAN*" <gaia.fiore@uniroma1.it>
Date: Tue, 24 Sep 2024 09:17:51 +0200
Message-ID: <CAHkmBVPVrn2rHkcg3EhhOfoA+1EHMLmUYYQ+G0KBmYM2ZBSPzA@mail.gmail.com>
Subject: QUESTA AZIONE È OBBLIGATORIA
To: undisclosed-recipients:;
Bcc: leonardo.querzoni@uniroma1.it
Content-Type: multipart/alternative; boundary="0000000000004e74070622d84ed1"

--0000000000004e74070622d84ed1
```

SIMPLE SPOOFING EXAMPLE 2

```
Delivered-To: querzoni@dis.uniroma1.it
Received: by 2002:a59:c26b:0:b0:48e:d422:933f with SMTP id c11csp2768947vqr;
        Tue, 24 Sep 2024 00:18:05 -0700 (PDT)
Return-Path: <leonardo.querzoni+caf_=querzoni=dis.uniroma1.it@uniroma1.it>
Received: from mail-sor-f41.google.com (mail-sor-f41.google.com. [209.85.220.41])
        by mx.google.com with SMTPS id 2adb3069b0e04-537a86516a3sor234209e87.25.2024.09.24.00.18.04
        for <querzoni@dis.uniroma1.it> (Google Transport Security);
        Tue, 24 Sep 2024 00:18:05 -0700 (PDT)
Return-Path: <gaia.fiore@uniroma1.it>
Received: from mail-sor-f41.google.com (mail-sor-f41.google.com. [209.85.220.41])
        by mx.google.com with SMTPS id 41be03b00d2f7-7e6b7c4586fsor419004a12.4.2024.09.24.00.18.02
        for <leonardo.querzoni@uniroma1.it> (Google Transport Security);
        Tue, 24 Sep 2024 00:18:03 -0700 (PDT)
MIME-Version: 1.0
From: "*SAPIENZAN*" <gaia.fiore@uniroma1.it>
Date: Tue, 24 Sep 2024 09:17:51 +0200
Message-ID: <CAHkmBVPVrn2rHkcg3EhhOfoA+1EHMLmUYYQ+G0KBmYM2ZBSPzA@mail.gmail.com>
Subject: QUESTA AZIONE È OBBLIGATORIA
To: undisclosed-recipients:;
Bcc: leonardo.querzoni@uniroma1.it
Content-Type: multipart/alternative; boundary="0000000000004e74070622d84ed1"

--0000000000004e74070622d84ed1
```

SIMPLE SPOOFING EXAMPLE 2

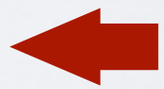
```
Delivered-To: querzoni@dis.uniroma1.it
Received: by 2002:a59:c26b:0:b0:48e:d422:933f with SMTP id c11csp2768947vqr;
        Tue, 24 Sep 2024 00:18:05 -0700 (PDT)
Return-Path: <leonardo.querzoni+caf_=querzoni=dis.uniroma1.it@uniroma1.it>
Received: from mail-sor-f41.google.com (mail-sor-f41.google.com. [209.85.220.41])
        by mx.google.com with SMTPS id 2adb3069b0e04-537a86516a3sor234209e87.25.2024.09.24.00.18.04
        for <querzoni@dis.uniroma1.it> (Google Transport Security);
        Tue, 24 Sep 2024 00:18:05 -0700 (PDT)
Return-Path: <gaia.fiore@uniroma1.it>
Received: from mail-sor-f41.google.com (mail-sor-f41.google.com. [209.85.220.41])
        by mx.google.com with SMTPS id 41be03b00d2f7-7e6b7c4586fsor419004a12.4.2024.09.24.00.18.02
        for <leonardo.querzoni@uniroma1.it> (Google Transport Security);
        Tue, 24 Sep 2024 00:18:03 -0700 (PDT)
MIME-Version: 1.0
From: "*SAPIENZAN*" <gaia.fiore@uniroma1.it>
Date: Tue, 24 Sep 2024 09:17:51 +0200
Message-ID: <CAHkmBVPVrn2rHkcg3EhhOfoA+1EHMLmUYYQ+G0KBmYM2ZBSPzA@mail.gmail.com>
Subject: QUESTA AZIONE È OBBLIGATORIA
To: undisclosed-recipients:;
Bcc: leonardo.querzoni@uniroma1.it
Content-Type: multipart/alternative; boundary="0000000000004e74070622d84ed1"

--0000000000004e74070622d84ed1
```

SIMPLE SPOOFING EXAMPLE 2

```
Delivered-To: querzoni@dis.uniroma1.it
Received: by 2002:a59:c26b:0:b0:48e:d422:933f with SMTP id c11csp2768947vqr;
        Tue, 24 Sep 2024 00:18:05 -0700 (PDT)
Return-Path: <leonardo.querzoni+caf_=querzoni=dis.uniroma1.it@uniroma1.it>
Received: from mail-sor-f41.google.com (mail-sor-f41.google.com. [209.85.220.41])
        by mx.google.com with SMTPS id 2adb3069b0e04-537a86516a3sor234209e87.25.2024.09.24.00.18.04
        for <querzoni@dis.uniroma1.it> (Google Transport Security);
        Tue, 24 Sep 2024 00:18:05 -0700 (PDT)
Return-Path: <gaia.fiore@uniroma1.it>
Received: from mail-sor-f41.google.com (mail-sor-f41.google.com. [209.85.220.41])
        by mx.google.com with SMTPS id 41be03b00d2f7-7e6b7c4586fsor419004a12.4.2024.09.24.00.18.02
        for <leonardo.querzoni@uniroma1.it> (Google Transport Security);
        Tue, 24 Sep 2024 00:18:03 -0700 (PDT)
MIME-Version: 1.0
From: "*SAPIENZAN*" <gaia.fiore@uniroma1.it>
Date: Tue, 24 Sep 2024 09:17:51 +0200
Message-ID: <CAHkmBVPVrn2rHkcg3EhhOfoA+1EHMLmUYYQ+G0KBmYM2ZBSPzA@mail.gmail.com>
Subject: QUESTA AZIONE È OBBLIGATORIA
To: undisclosed-recipients:;
Bcc: leonardo.querzoni@uniroma1.it
Content-Type: multipart/alternative; boundary="0000000000004e74070622d84ed1"

--0000000000004e74070622d84ed1
```



SECURITY CHALLENGES

Most of these threats stem from the lack of adequate security guarantees in the standard email architecture and its protocols

- No sender authentication means no way to trace the origin of an email
- No end-to-end encryption means no way to enforce content confidentiality

SENDER AUTHENTICATION

Sender Policy Framework (SPF)

- Verifies if the sending server is authorized to send emails on behalf of the domain - RFC 4408

DomainKeys Identified Mail (DKIM)

- Provides a cryptographic signature to ensure the email's integrity and authenticity - RFC 4871

Domain-based Message Authentication, Reporting & Conformance (DMARC)

- Enforces policies based on SPF and DKIM and sends reports back to domain owners on authentication failures.

Authenticated Receiver Chain (ARC)

- Allows for maintaining the validity of DMARC policies across intermediaries.

SENDER POLICY FRAMEWORK

- **SPFv1 (or SPF Classic)** protects the sender address (in envelope) by allowing the owner of a domain to specify a mail sending policy, namely which mail servers are authorized to send mail from the domain, using special DNS records (SPF, type 99)
- If server accepts the sender, recipients and body of message, it should insert a Return-Path field in the message header in order to save the sender address
  - While the address in the Return-Path often matches other originator addresses in the mail header such as From or Sender, this is not necessarily the case, and SPF does not prevent forgery of these other addresses
- Guarntees Sender Server Authorization against mail spoofing.

SPF BY EXAMPLE

- Bob owns domain *example.net*.
- He also sometimes sends mail through his GMail account and contacted GMail's support to identify the correct SPF record for GMail.
- Since he often receives bounces about messages he didn't send, he decides to publish an SPF record in order to reduce the abuse of his domain in e-mail envelopes:

```
example.net  TXT  "v=spf1 mx a:pluto.example.net include:aspmx.googlemail.com -all"
```

- Check spf record

```
❯ dig +noall +answer uniroma1.it txt
```

SPF BY EXAMPLE

```
example.net  TXT  "v=spf1 mx a:pluto.example.net include:aspmx.googlemail.com -all"
```

SPF record item	Description
`v=spf1`	SPF version 1
`mx`	The incoming mail servers (MXes) of the domain are authorized to also send mail for example.net
`a:pluto.example.net`	the machine pluto.example.net is authorized, too
`include:aspmx.googlemail.com`	every mail server considered legitimate by gmail.com is legitimate for example.net
`~all`	all other servers are not authorized, and email they generate should be marked as suspicious

SPF BY EXAMPLE

- When an email is sent from the domain, the receiving mail server queries the DNS for the domain's SPF record to verify if the IP address of the sending server is listed in the SPF record.
  - **Pass**: If the sending server's IP matches one of the authorized IPs in the SPF record, the email passes the SPF check.
  - **Fail**: If the sending IP is not listed, the email fails the SPF check. The receiving server can then choose to reject, flag, or accept the message based on local policy.

```
<XXXX.YYYY@gmail.com>: host gmail-smtp-in.l.google.com[173.194.78.26]
said: 550-5.7.1 [aa.bb.cc.dd] The IP you're using to send mail is not
authorized to 550-5.7.1 send email directly to our servers. Please use the
SMTP relay at your 550-5.7.1 service provider instead. Learn more at 550
5.7.1 http://support.google.com/mail/bin/answer.py?answer=10336
fl4si3665795wib.12 - gsmtp (in reply to end of DATA command)
```

SPF BY EXAMPLE

MX TOOLBOX® SUPERTOOL

Pricing

SuperTool | MX Lookup | Blacklists | DMARC | Diagnostics | Email Health | DNS Lookup | Analyze Headers

SuperTool Beta9

uniroma1.it | SPF Record Lookup

spf:uniroma1.it Find Problems | Solve Email Delivery Problems

Gmail & Yahoo are now requiring DMARC - Get yours setup with Delivery Center

```
v=spf1 ip4:151.100.101.67 ip4:151.100.101.143 ip4:130.186.31.160/27 IP4:130.186.7.107 include:_spf.google.com -all
```

Prefix	Type	Value	PrefixDesc	Description
	v	spf1		The SPF record version
+	ip4	151.100.101.67	Pass	Match if IP is in the given range.
+	ip4	151.100.101.143	Pass	Match if IP is in the given range.
+	ip4	130.186.31.160/27	Pass	Match if IP is in the given range.
+	IP4	130.186.7.107	Pass	Match if IP is in the given range.
+	include	_spf.google.com	Pass	The specified domain is searched for an 'allow'.
-	all		Fail	Always matches. It goes at the end of your record.

SPF BY EXAMPLE

MX TOOLBOX®
SUPERTOOL

Pricing Tools

SuperTool | MX Lookup | Blacklists | DMARC | Diagnostics | Email Health | DNS Lookup | Analyze Headers

SuperTool Beta9

uniroma1.it MX Lookup

mx:uniroma1.it Find Problems | Solve Email Delivery Problems

Pref	Hostname	IP Address	TTL		
1	ASPMX.L.GOOGLE.COM	142.251.163.27 Google LLC (AS15169)	24 hrs	Blacklist Check	SMTP Test
1	ASPMX.L.GOOGLE.COM	2607:f8b0:4004:c17::1a	24 hrs	Blacklist Check	
5	ALT1.ASPMX.L.GOOGLE.COM	209.85.202.27 Google LLC (AS15169)	24 hrs	Blacklist Check	SMTP Test
5	ALT1.ASPMX.L.GOOGLE.COM	2a00:1450:400b:c00::1b	24 hrs	Blacklist Check	
5	ALT2.ASPMX.L.GOOGLE.COM	64.233.184.27 Google LLC (AS15169)	24 hrs	Blacklist Check	SMTP Test
5	ALT2.ASPMX.L.GOOGLE.COM	2a00:1450:400c:c0b::1a	24 hrs	Blacklist Check	
10	ASPMX2.GOOGLEMAIL.COM	209.85.202.26 Google LLC (AS15169)	24 hrs	Blacklist Check	SMTP Test
10	ASPMX2.GOOGLEMAIL.COM	2a00:1450:400b:c00::1a	24 hrs	Blacklist Check	
10	ASPMX3.GOOGLEMAIL.COM	64.233.184.26 Google LLC (AS15169)	24 hrs	Blacklist Check	SMTP Test
10	ASPMX3.GOOGLEMAIL.COM	2a00:1450:400c:c0b::1a	24 hrs	Blacklist Check	

SPF LIMITATIONS

Email forwarding

- When an email is forwarded, the SPF check can fail.
  - The forwarder's IP address may not be included in the original sender's SPF record

Shared or third-party email services

- Organizations often use third-party services (like CRM platforms, marketing tools, or cloud services) to send emails on their behalf. These services may not be listed in the domain's SPF record.

Mailing lists

- Emails sent through mailing lists often pass through multiple servers before reaching recipients. Mailing list software sometimes modifies the email headers, including the sender's information, which can break SPF validation.

SPF LIMITATIONS

SPF record size limitations

- The size limit for an SPF record is 450 characters minus the length of the domain name and the length of any other TXT record value. This is the upper exclusive limit recommended in RFC 7208, the document that specifies SPF.

No protection for email body

- SPF only authenticates the sending server's IP address and does not protect the email's content or prevent modification of the message body or attachments.

No full sender verification

- SPF only verifies the sending server's IP address against the envelope sender (the "MAIL FROM" address), not the From: header that the user sees.

SPF LIMITATIONS

Misconfiguration

- Incorrectly configured SPF records, such as syntax errors or forgetting to include authorized sending IPs, can cause legitimate emails to fail SPF checks.

Complex mail routing

- In some scenarios with complex email routing (e.g., email relays, hybrid systems), SPF checks can fail because the originating IP doesn't match the IP specified in the SPF record.

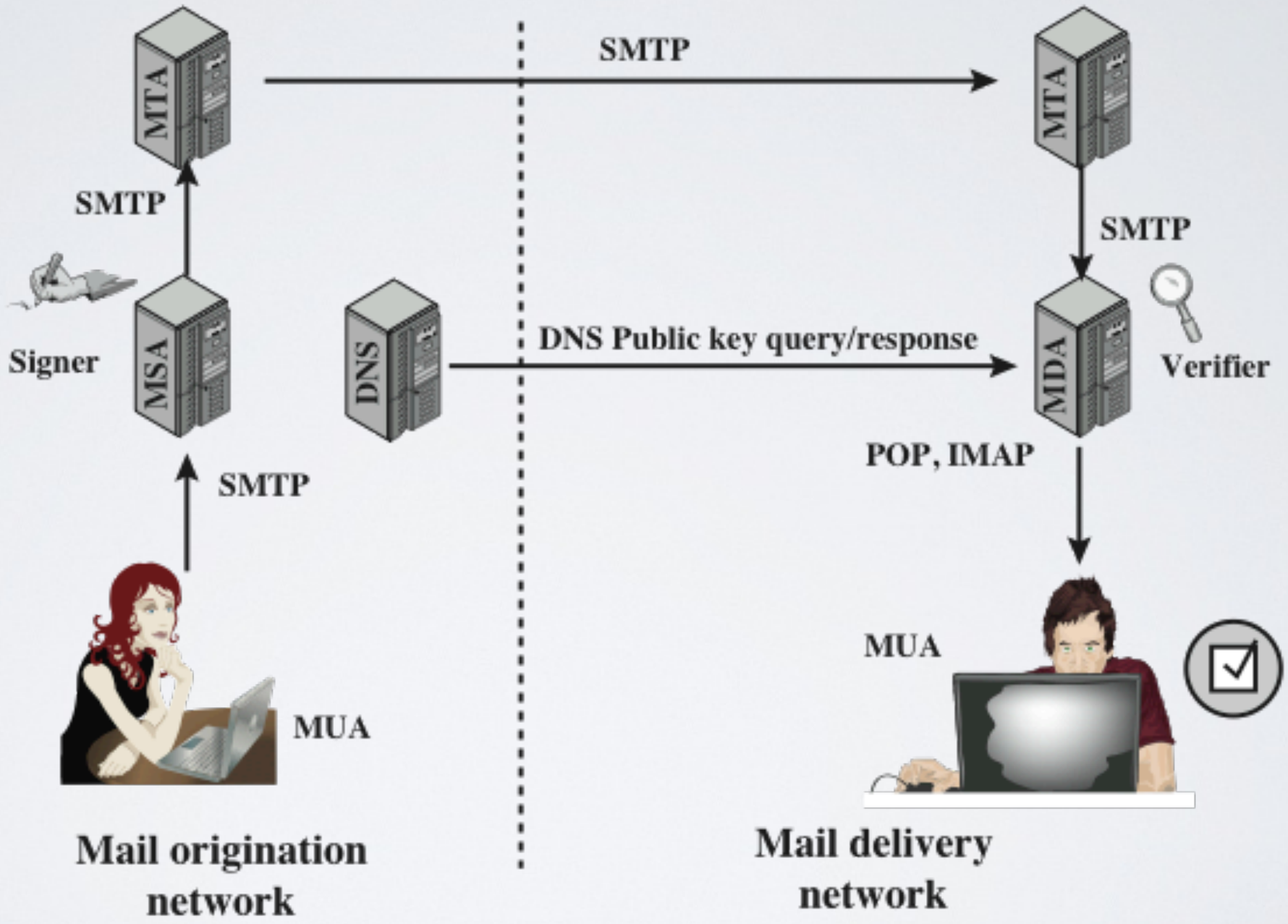
Outbound-Only Security

- SPF is mainly used to authenticate outbound email servers, but it doesn't offer any protection or guarantee about the inbound email a domain receives.

DKIM

- **DomainKeys Identified Mail (DKIM)** - RFC 4871 - is a specification for cryptographically signing e-mail messages, permitting a signing domain to claim responsibility for a message.
- Message recipients (or agents acting in their behalf) can verify the signature by querying the signer's domain to retrieve the public key
- What DKIM guarantees:
  - Email content integrity
  - Domain authentication
  - Non repudiation

POSSIBLE DKIM DEPLOYMENT



HOW DOES DKIM WORK

1 - Generating the Signature (Signing Process):

- When a domain sends an email, the sender's mail server (or an intermediary) generates a cryptographic signature based on specific parts of the message (e.g., the body and certain headers, like the subject or the From field). The signing is done using a private key that only the domain owner controls.
- The signature is added to the email as a new header: the DKIM-Signature header. This signature contains information like:
  - The hashing algorithm used.
  - The domain that is taking responsibility for the email.
  - The email parts that were signed (headers and body).
  - The signature itself (a base64-encoded value).

HOW DOES DKIM WORK

1 - Generating the Signature (Signing Process):

```
DKIM-Signature:
        v=1;
        a=rsa-sha256;
        c=relaxed/relaxed;
        d=diag.uniroma1.it;
        s=google;
        t=1728828422;
        x=1729433222;
        dara=google.com;
        h=to:subject:message-id:date:from:mime-version:from:to:cc:subject:date:message-
        id:reply-to;
        bh=t6DYHnvgeJvZ02sgmWVU/4X9LTVieRyKbl+FRWPu3Co=;
        b=gx0VlcD55ZtEOTnE2FJJSSgt8Padr87pkhbkWw7FbuIyWY2O0QvKy52DD6DsuiT2oj
          q5/jSGM4y1b0XKHM7CU1Fhk+ScKi16hj8HdezlpnFOZbbiKS43uysV8lLfbv5SOaDEFv
          REmsa4Az8duRs1fYEsQ9ixRu5RPOLRgCBcxb0=
```

HOW DOES DKIM WORK

2 - Public Key Published in DNS:

- The domain that sends the email publishes the corresponding public key in the Domain Name System (DNS) as a TXT record. The public key is used to verify the authenticity of the signature.
- The DNS record also specifies the selector (a prefix to differentiate between multiple keys) and the policy the sender wants to use for DKIM.

```
› dig +noall +answer google._domainkey.uniroma1.it txt
google._domainkey.uniroma1.it. 21600 IN    TXT "v=DKIM1; k=rsa;
p=MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAzMa8wGDtu7DVjVP1JwVzMym/
KktdVSBhvtMbgpolQTWqKxRHejICsUvvFv6WGP7kKQnVA5" "2JtFU9LVGvTfkNF5J/x/
9wUlBSQMCwGc4IXNdgA5fcn/49fV+YY1RFY44PhoTSWTQnKp7axDRFO3Uo05uFXSl0OnNo0Gd/
tnDRG538tnM8VzZIF+jjS76GkV/iZT2tcDSBMsWjZTR" "tk7eG/GDVS8pbD14CX/
bf7RTflt3sTiwcp3YUn5T66ioCmc7PIu5CCKfTcW7i7E246Ef4hz+6CySyNRxipnrK6BrXGoraod5U66K6boXVW
ojDKHRflvdoeQ49hW8N5PHFqJKebwIDAQAB"
```

SELECTORS

To support multiple concurrent public keys per signing domain, key namespace is subdivided using selectors

- for example selectors might indicate the names of office locations, the signing date, or even the individual user

Selectors are useful to implement some important use cases

- domains that want to delegate signing capability for a specific address for a given duration to a partner, such as an advertising provider or other outsourced function
- domains that want to allow frequent travelers to send messages locally without the need to connect with a particular MSA.
- "affinity" domains (e.g., college alumni associations) that provide forwarding of incoming mail, but that do not operate a MSA for outgoing mail

HOW DOES DKIM WORK

3 - Verification Process:

- When an email is received, the recipient's mail server looks at the DKIM signature header to see which domain signed the message and which selector to use. It retrieves the corresponding public key from the domain's DNS.
- The server then verifies the digital signature by comparing it with the hash of the received message's content. If the signature matches, it confirms that the email:
  - Hasn't been altered in transit (message integrity).
  - Is indeed authorized by the sender's domain (authenticity).

HOW DOES DKIM WORK

3 - Passing or Failing the DKIM Check:

- **Pass**: If the signature is valid, the message is authenticated, and the recipient server knows that the message came from an authorized source and hasn't been tampered with.
- **Fail**: If the signature doesn't match (due to changes in the content during transit) or if the public key in DNS doesn't match, the message fails the DKIM check.
- The final action (e.g., rejection, quarantine) depends on other factors (e.g. DMARC policy).

DKIM AND SPF

How does DKIM compares to SPF?

- SPF authenticates the sending server's IP address against the domain's SPF record.
- DKIM authenticates the email content by verifying the signature against a public key stored in DNS.

Limitations

- DKIM doesn't authenticate the visible From address. It only verifies the email's authenticity from the domain that signed the message.
- Email forwarding can break DKIM if the forwarding server alters the content (e.g., adding disclaimers).
- It doesn't provide encryption or privacy—only authenticity and integrity.

CANONICALIZATION

- e-mail servers and relay systems may modify email in transit, potentially invalidating a signature
- headers are subjected to a canonicalization algorithm
  - **relaxed** (tolerating) or **simple** (strict)
- bodies are also subjected to a canonicalization algorithm
  - choices for header/body are independent
- see RFC 4871 for details

DKIM EXAMPLE

```
DKIM-Signature:
    v=1;
    a=rsa-sha256;
    c=relaxed;
    d=example.com;
    s=mail;
    h=From:To:Subject:Date;
    bh=MTIzNDU2Nzg5MDEyMzQ1Njc4OQ==;
    b=abcdefghijklmnopqrstuvwxyz1234
    567890abcdefghijklmnopqrstuvwxyz
```

a = Hash/signing algorithm

q = Algorithm for getting public key

d = Signing domain

i = Signing identity

s = Selector

c = Canonicalization algorithm (*relaxed*/*simple*)

t = Signing time (seconds since 1/1/1970)

x = Expiration time

h = List of headers included in signature; *DKIM-Signature* is implied

b = The signature itself

bh = The hash of the canonicalized body part of the message

v = DKIM version

DMARC

- **Domain-based Message Authentication, Reporting, and Conformance**, is a technical standard (RFC 7489) that helps protect email senders and recipients from spam/spoofing/phishing.
- DMARC allows an organization to publish a policy that defines its email authentication practices and provides instructions to receiving mail servers for how to enforce them.
- Specifically, DMARC establishes a method for a domain owner to:
  - Publish its email authentication practices
  - State what actions should be taken on mail that fails authentication checks
  - Enable reporting of these actions taken on mail claiming to be from its domain
- Puts together SPF and DKIM

HOW DOES DMARC WORK?

1) A domain administrator publishes the policy defining its email authentication practices and how receiving mail servers should handle mail that violates this policy.

- This DMARC policy is listed as part of the domain's overall DNS records.
- A DMARC record is included in an organization's DNS database. It is a specially-formatted version of a standard DNS TXT record with a particular name: `_dmarc.mydomain.com`

HOW DOES DMARC WORK?

DMARC record example

```
› dig +noall +answer _dmarc.uniroma1.it txt
_dmarc.uniroma1.it.   21600    IN   TXT "v=DMARC1; p=reject; pct=10;
rua=mailto:lxoyu6mk@ag.eu.dmarcadvisor.com,mailto:dmarc-ar@uniroma1.it;
ruf=mailto:lxoyu6mk@fr.eu.dmarcadvisor.com,mailto:dmarc-f@uniroma1.it;"
```

- **`v=DMARC1`** specifies the DMARC version
- **`p=reject`** specifies the preferred treatment, or DMARC policy
- **`rua=mailto:…`** is the mailbox to which aggregate reports should be sent
- **`ruf=mailto:…`** is the mailbox to which forensic reports should be sent
- **`pct=10`** is the percentage of mail to which the domain owner would like to have its policy applied

HOW DOES DMARC WORK?

DMARC record example

```
› dig +noall +answer _dmarc.uniroma1.it txt
_dmarc.uniroma1.it.   21600    IN   TXT "v=DMARC1; p=reject; pct=10;
rua=mailto:lxoyu6mk@ag.eu.dmarcadvisor.com,mailto:dmarc-ar@uniroma1.it;
ruf=mailto:lxoyu6mk@fr.eu.dmarcadvisor.com,mailto:dmarc-f@uniroma1.it;"
```

The DMARC specification provides three choices for domain owners to specify their preferred treatment of mail that fails DMARC validation checks. These policies are:

- **none**: treat the mail the same as it would be without any DMARC validation
- **quarantine**: accept the mail but place it somewhere other than the recipient's inbox (typically the spam folder)
- **reject**: reject the message outright

HOW DOES DMARC WORK?

2) When an inbound mail server receives an incoming email, it uses DNS to look up the DMARC policy for the domain contained in the message's "From" (RFC 5322) header.

The inbound server then checks the message for three key factors:

- Does the message's DKIM signature validate?
- Did the message come from IP addresses allowed by the sending domain's SPF records?
- Do the headers in the message show proper "domain alignment"?

HOW DOES DMARC WORK?

2) When an inbound mail server receives an incoming email, it uses DNS to look up the DMARC policy for the domain contained in the message's "From" (RFC 5322) header.

"Domain alignment" is a concept in DMARC that expands the domain validation intrinsic to SPF and DKIM. DMARC domain alignment matches a message's "from" domain with information relevant to these other standards:

- For SPF, the message's From domain and its Return-Path domain must match
- For DKIM, the message's From domain and its DKIM d= domain must match

HOW DOES DMARC WORK?

3)With this information, the server is ready to apply the sending domain's DMARC policy to decide whether to accept, reject, or otherwise flag the email message.

4) After using DMARC policy to determine the proper disposition for the message, the receiving mail server will report the outcome to the sending domain owner.

HOW DOES DMARC WORK?

DMARC reports are generated by inbound mail servers as part of the DMARC validation process. There are two formats of DMARC reports:

- **Aggregate reports**, which are XML documents showing statistical data about the messages received that claimed to be from a particular domain. Data reported includes authentication results and message disposition. Aggregate reports are designed to be machine-readable.
- **Forensic reports**, which are individual copies of messages which failed authentication, each enclosed in a full email message using a special format called AFRF. Forensic report can be useful both for troubleshooting a domain's own authentication issues and for identifying malicious domains and web sites.

DMARC CHECK OUTCOMES

Outcomes from protocol checks are reported in the mail headers

```
Authentication-Results: mx.google.com;
       dkim=pass header.i=@diag.uniroma1.it header.s=google header.b=gx0VlcD5;
       spf=pass (google.com: domain of querzoni@diag.uniroma1.it designates
      209.85.220.41 as permitted sender) smtp.mailfrom=querzoni@diag.uniroma1.it;
       dmarc=pass (p=NONE sp=NONE dis=NONE) header.from=diag.uniroma1.it;
       dara=pass header.i=@gmail.com
```

DMARC LIMITATIONS

Domains with strict DMARC policies (p=reject) may see legitimate messages blocked if they go through indirect mailflows such as mailing lists, forwarding, or filtering services

- Forwarding causes SPF to fail even if origin was legit
- Forwarders often alter messages, breaking DKIM
- Disclaimers and footers
- Virus scan results
- Removed attachments
- Mailing list subject tags

Some text for ARC-related slides is from The Trusted Domain Project

DMARC LIMITATIONS

Example:

- Intermediary sends the message from a new IP address, causing SPF to fail to verify for Sender's domain
- Intermediary changes the message contents, causing Sender's DKIM signature to fail to verify

AUTHENTICATED RECEIVED CHAIN

ARC helps preserve the authentication status of an email that passes through multiple intermediaries or forwarding services.

Guarantees:

- **Preserves Authentication**: ARC ensures that if an email passes SPF, DKIM, and DMARC checks at the original source, this authenticated status can be carried forward to subsequent servers or recipients.
- **Accountability for Intermediaries**: Each intermediary in the email flow adds its own ARC authentication results, creating a verifiable chain of who has handled the message and whether they altered the original email.
- **Improved Email Deliverability**: ARC can improve deliverability for emails sent through mailing lists, email forwarding services, or resending platforms by helping them pass DMARC checks that would otherwise fail.

AUTHENTICATED RECEIVED CHAIN

ARC helps preserve the authentication status of an email that passes through multiple intermediaries or forwarding services.

- **Preserves Authentication**: ARC ensures that if an email passes SPF, DKIM, and DMARC checks at the original source, this authenticated status can be carried forward to subsequent servers or recipients.
- **Accountability for Intermediaries**: Each intermediary in the email flow adds its own ARC authentication results, creating a verifiable chain of who has handled the message and whether they altered the original email.
- **Improved Email Deliverability**: ARC can improve deliverability for emails sent through mailing lists, email forwarding services, or resending platforms by helping them pass DMARC checks that would otherwise fail.

ARC does not replace DMARC but works alongside it. ARC is concerned with **preserving the email's authentication chain**.

AUTHENTICATED RECEIVED CHAIN

What ARC does not do:

- Does not say anything about “trustworthiness” of the message sender or intermediaries
- Says nothing about the contents of the message
- Intermediaries might still inject bad content
- Intermediaries might remove some or all ARC headers

AUTHENTICATED RECEIVED CHAIN

ARC introduces three header fields:

`ARC-Authentication-Results:`(AAR) Archived copy of Authentication-Results:. Records the results of any authentication checks (like SPF or DKIM) performed by the intermediary.

`ARC-Seal:`(AS) Includes a DKIM-style cryptographic signature that verifies the authenticity of the ARC chain up to this point, confirming that no one has tampered with the previous ARC headers.

`ARC-Message-Signature:`(AMS) A DKIM-style signature of the entire message except ARC-Seal: headers. Helps preserving the integrity of the message as it passes through intermediaries.

AUTHENTICATED RECEIVED CHAIN

The `ARC-Seal:` header contains the following fields:

- `b=` is a signature of all ARC headers; no non-ARC headers
- `a=/d=/s=` fields match the corresponding DKIM tags
  - Same key format and DNS records as for DKIM
  - Can use your DKIM keys for ARC
  - Can use separate keys per local policy or preference
- `cv=` indicates whether ARC chain validated as received by the reporting intermediary
- `i=` tag is a sequence number for ARC header sets

AUTHENTICATED RECEIVED CHAIN

To sign a modification, an intermediate server performs the following steps:

1. The content of `Authentication-Results:` content is copied into a new `ARC-Authentication-Results:` header, prefixed to the message
2. Calculates the `ARC-Message-Signature:` for the message, including the latest AAR header, and prefixed to the message
   - Must not include any `ARC-Seal:` headers
3. `ARC-Seal:` is calculated and prefixed

ARC headers are prefixed per common practice, but order of appearance is not critical for validation

NOTE: a new ARC header must be inserted ONLY if the intermediary makes changes that may break DMARC checks.

AUTHENTICATED RECEIVED CHAIN

The `i=` sequence tag is used to order the ARC headers for various operations:

- Allows multiple ARC header sets to be grouped easily and correctly
- Eliminates reliance on the order of headers being inserted – or not being altered
- Compare with order of insertion of various authentication, content scanning, or `Received:` headers

AUTHENTICATED RECEIVED CHAIN

Steps to check an ARC chain:

- Verify the `ARC-Seal:` — Ensure the cryptographic signature in the seal matches the domain and selector used by the intermediate server.
- Verify the `ARC-Message-Signature:` — Check that the signature in the ARC-Message-Signature correctly matches the email's original content and headers.
- Check `Authentication Results:` — Review the ARC-Authentication-Results to see if SPF, DKIM, and other checks passed for the original email.
- Evaluate the Chain — Continue verifying the chain of ARC headers through all the intermediaries. Each hop adds another ARC-Seal, which must be validated to maintain the integrity of the chain.

AUTHENTICATED RECEIVED CHAIN

Example of ARC Headers:

```
ARC-Seal: i=3; a=rsa-sha256; t=1608562589; cv=pass;
      d=google.com; s=arc-20160816;
      b=eDQJjq5aDJrTBzWbJU0dt46mObilIR3UFyYJcr6lYC0g3n3EosZUStgpmuSXSV1[…]

ARC-Message-Signature: i=3; a=rsa-sha256; c=relaxed/relaxed; d=google.com;
      s=arc-20160816;
      h=list-unsubscribe:list-archive:list-help:list-post:list-id […]
      bh=C+v3YGJvRDYuOojq0nvM9h7NPqS2uE+OtdOmVd6jGOc=;
      b=cWvAcmX8L3q3iNVeBQKsgYT97hqqbMHFDK/E1cSrjjdIX2yMTJ7m5aYo0ldvECo […]

ARC-Authentication-Results: i=3; mx.google.com;
      dkim=pass header.i=@ceitnet.it header.s=google header.b="JVoNyZ/W";
      arc=pass (i=2 spf=pass spfdomain=persiplast.com.br dkim=pass
      dkdomain=persiplast.com.br);
       spf=pass (google.com: domain of delivery.of.????@ceitnet.it designates
          209.85.220.69 as permitted sender) smtp.mailfrom=delivery.of.????@ceitnet.it
```

END-TO-END SECURITY

For a fully-trustable email system we would like the following guarantees to hold for ANY message:

- confidentiality of message content
- authentication of message sender
- message integrity
- sender non-repudiation

SECURING E-MAIL BY PGP

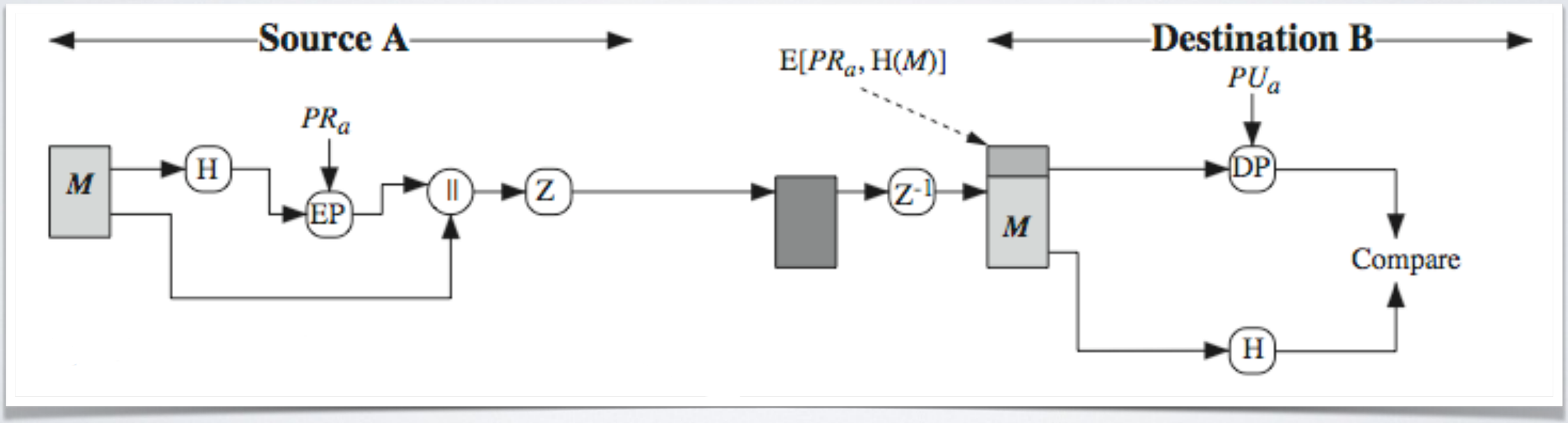
- Pretty Good Privacy is a standard created by Phil Zimmermann in 1991
  - "PGP empowers people to take their privacy into their own hands. There has been a growing social need for it. That's why I wrote it." See Why I wrote PGP https://www.philzimmermann.com/EN/essays/WhyIWrotePGP.html
- The slides on PGP are inspired to the well-known textbook Cryptography and Network Security, 5/e, by William Stallings, Chapter 18 – "Electronic Mail Security"

PRETTY GOOD PRIVACY (PGP)

- Well known and widely used since the 90s
- Using best available crypto algorithms
- Integrated into a single program
  - Linux/Unix, PC, Macintosh and other systems
- Originally free, now owned by Symantec (www.pgp.com)
- open version (OpenPGP) standardized in RCF 4880
  - several implementations, e.g., Gnu Privacy Guard (www.gnupg.org)

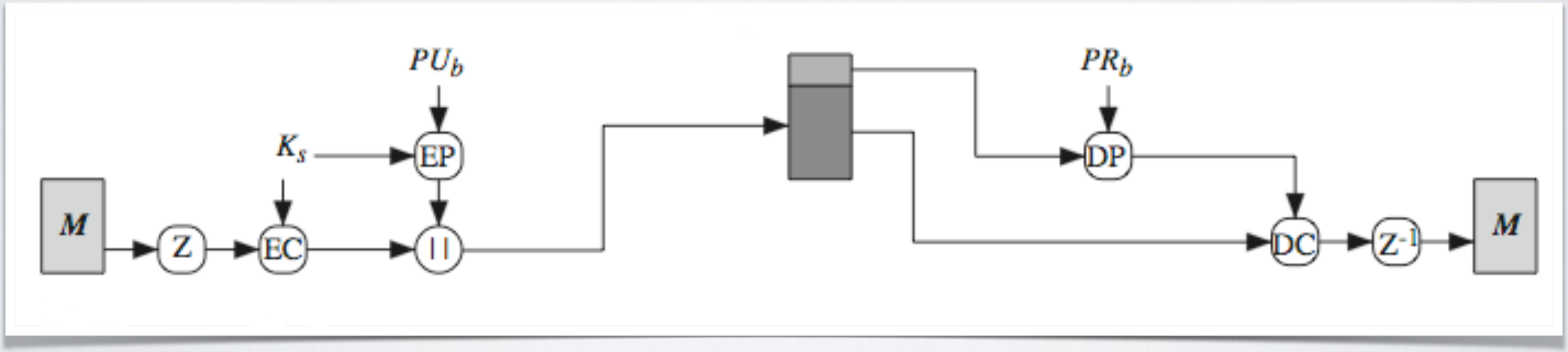
PGP AUTHENTICATION

1. sender creates message
2. make SHA-1 160-bit hash of message
3. attached RSA signed hash to message
4. receiver decrypts & recovers hash code
5. receiver verifies received message hash



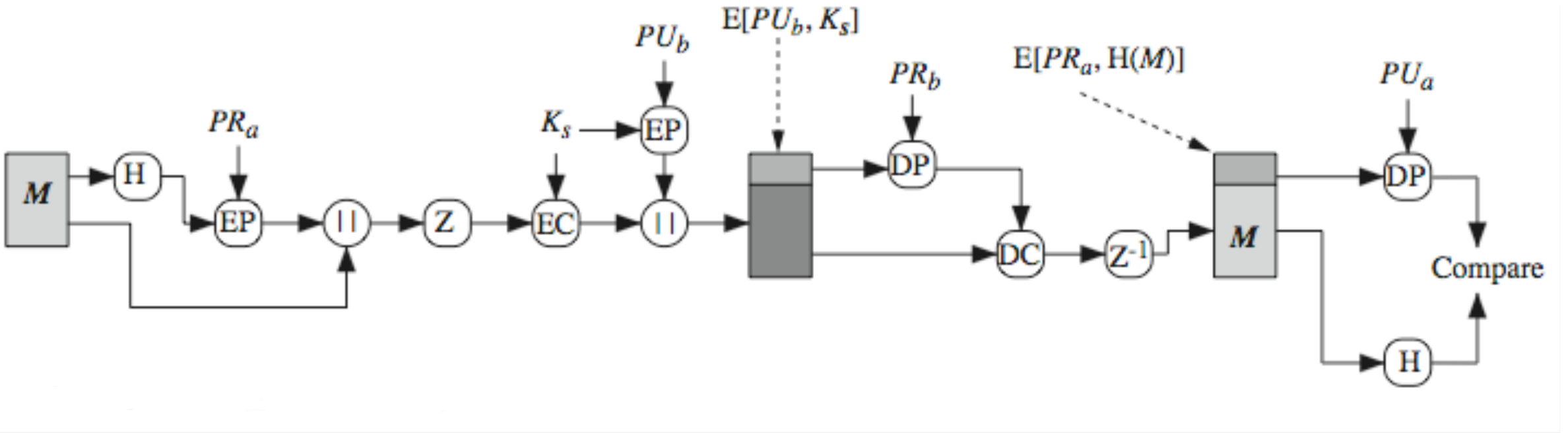
PGP CONFIDENTIALITY

1. sender forms 128-bit random session key
2. encrypts message with session key
3. attaches session key encrypted with RSA
4. receiver decrypts & recovers session key
5. session key is used to decrypt message



CONFIDENTIALITY & AUTHENTICATION

- can use both services on same message
  - create signature & attach to message
  - encrypt both message & signature
  - attach RSA/El-Gamal encrypted session key



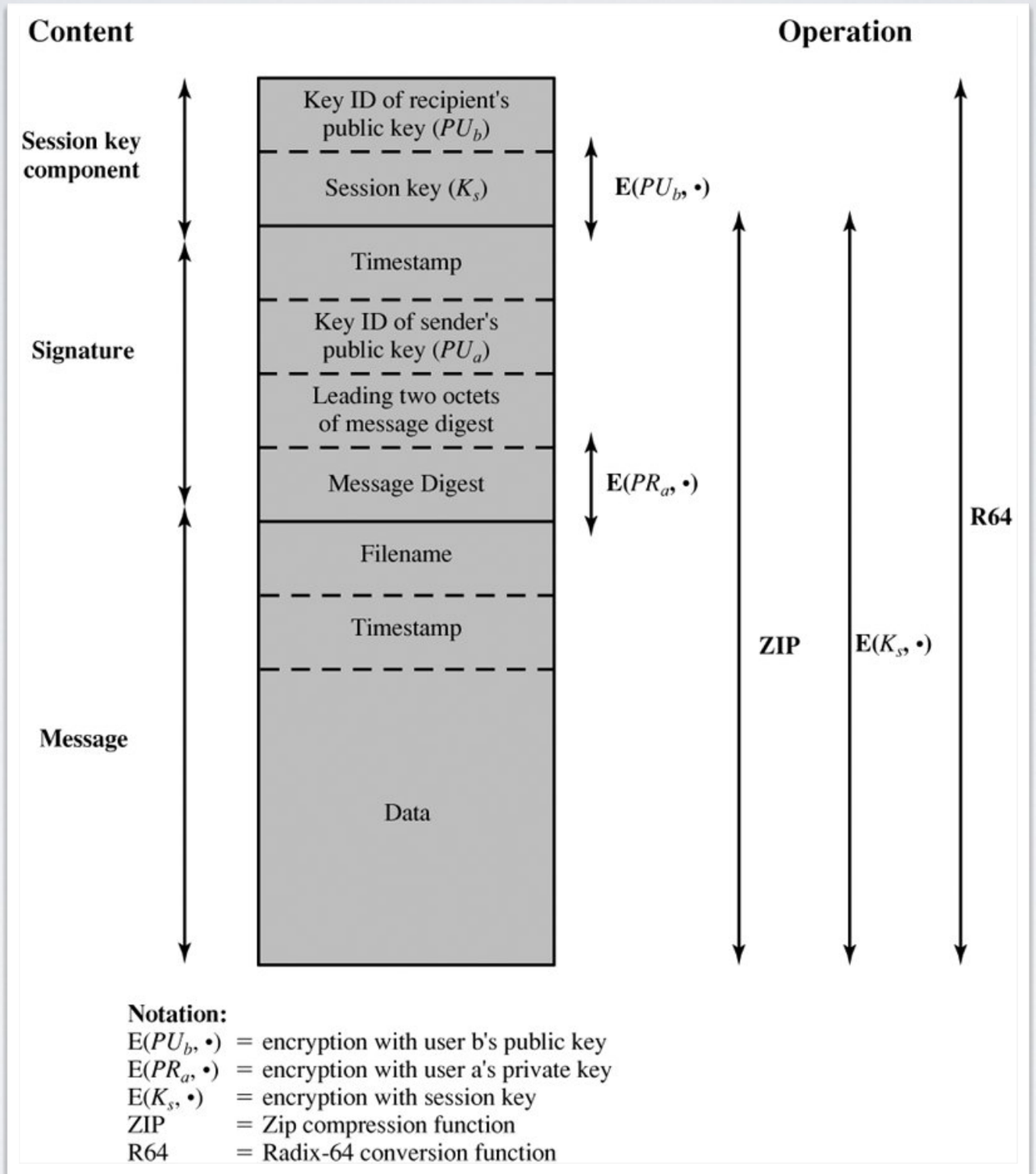
COMPRESSION

- by default PGP compresses message after signing but before encrypting
  - so can store uncompressed message & signature for later verification
  - because compression is non deterministic (if verification requires compression it may fail on legitimate messages)
- uses ZIP compression algorithm

PGP PUBLIC & PRIVATE KEYS

- since many public/private keys may be in use (by one user), need to identify which is actually used to encrypt session key in a message
  - could send full public-key with every message
  - but this is inefficient
- rather use a key identifier (ID) based on key
  - least significant 64-bits of the key
  - will very likely be unique
- also use key ID in signatures

PGP MESSAGE FORMAT



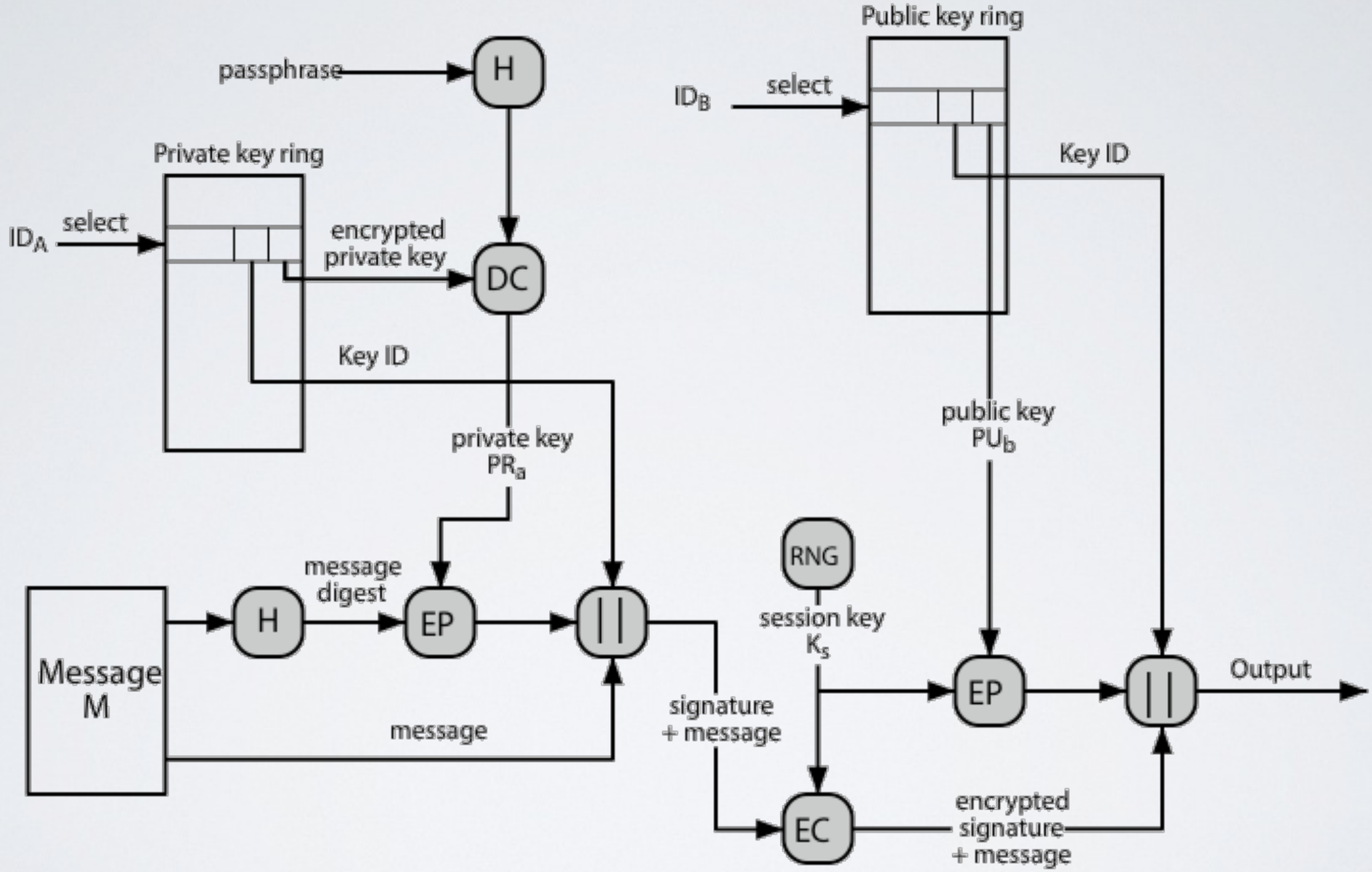
PGP KEY RINGS

Each PGP user has a pair of keyrings:

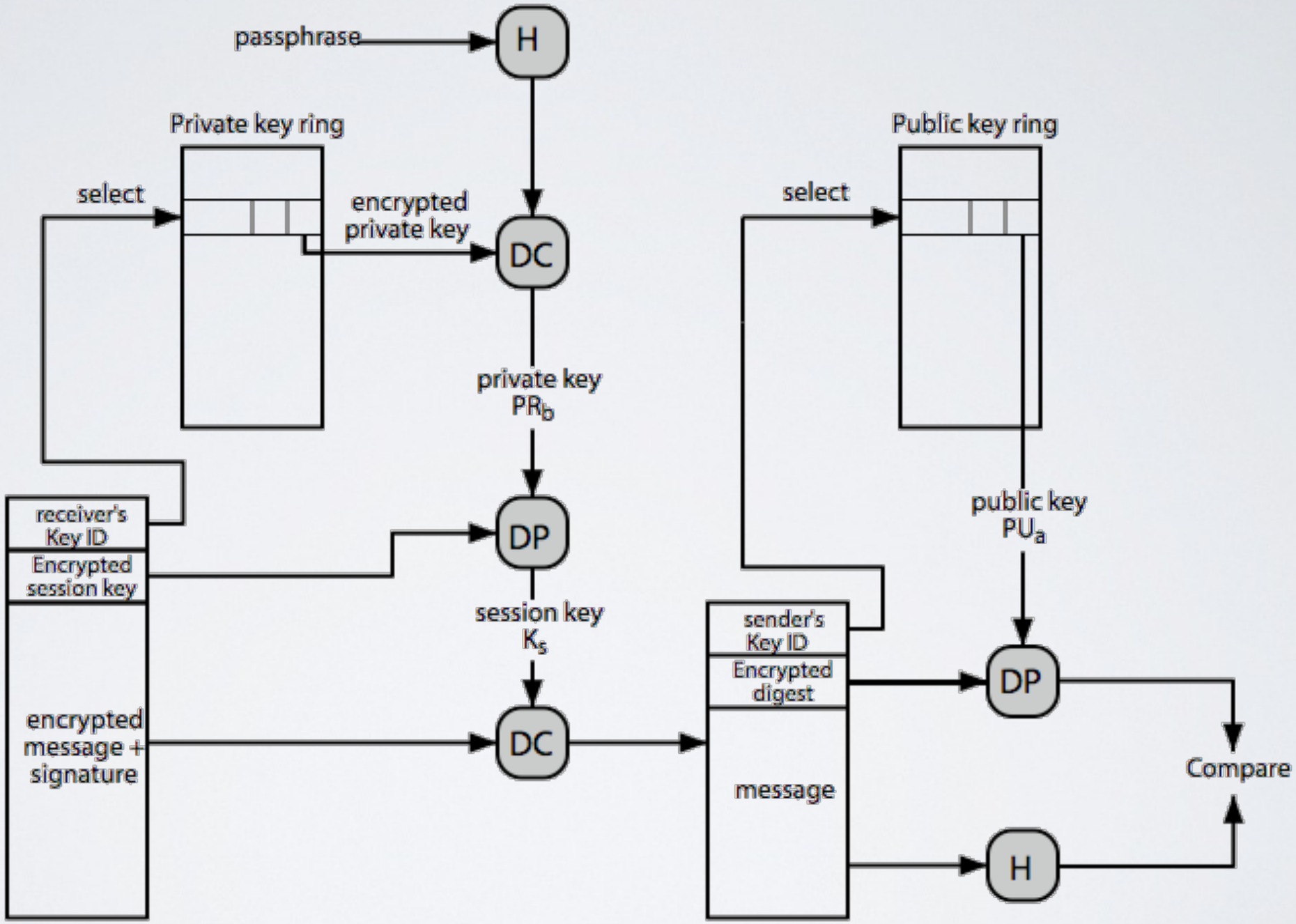
- **public-key ring** contains all the public-keys of other PGP users known to this user, indexed by key ID
- **private-key ring** contains the public/private key pair(s) for this user, indexed by key ID & encrypted keyed from a hashed passphrase

Security of private keys thus depends on the passphrase security

PGP MESSAGE GENERATION



PGP MESSAGE RECEPTION



PGP KEY MANAGEMENT

rather than relying on certificate authorities in PGP every user is his own CA

- can sign keys for users they know directly

The idea is to create a “web of trust”

- trust keys are signed
- can trust keys others have signed if have a chain of signatures to them

key ring includes trust indicators

- users can also revoke their keys

a possible key-sign procedure http://herrons.com/keysigning-party-guide/

WEB OF TRUST (ZIMMERMANN)

“As time goes on, you will accumulate keys from other people that you may want to designate as trusted introducers. Everyone else will each choose their own trusted introducers. And everyone will gradually accumulate and distribute with their key a collection of certifying signatures from other people, with the expectation that anyone receiving it will trust at least one or two of the signatures. This will cause the emergence of a decentralized fault-tolerant web of confidence for all public keys.”

S/MIME

- Introduces MIME extension for e-mail security
- Provides the same guarantees as PGP
- Different model that uses a centralised PKI for key management
- Uses X.509 certificates instead of plain public/private key pairs
- Less user control
- Better interoperability