HTTP Public Key Pinning (HPKP)

HTTP Public Key Pinning (HPKP) was **a web security mechanism that allowed website administrators to specify which public keys should be trusted for their domain**. It aimed to protect against attacks such as man-in-the-middle (MITM) or rogue certificate issuance by certificate authorities (CAs). However, HPKP has since been deprecated due to implementation challenges and potential risks.

1. How HPKP Worked

HPKP was **implemented via the Public-Key-Pins HTTP response header**. This header informed browsers about the public keys that should be pinned for the domain.

Key Components

- Public-Key-Pins Header
    - The header includes
        - A list of hashes of public keys that are allowed for the domain.
        - A max-age directive, specifying how long the pins should be cached.
        - An optional report-uri directive to report pin validation failures.
- Example HPKP Header

```
Public-Key-Pins:
    pin-sha256="base64==";
    pin-sha256="backup-base64==";
    max-age=5184000;
    includeSubDomains;
    report-uri="https://example.com/hpkp-report";
```

- Explanation
    - pin-sha256: Base64-encoded hash of the public key(s).
    - max-age: Specifies the duration (in seconds) the browser should enforce the policy (e.g., 5184000 = 60 days).
    - includeSubDomains: Extends the pinning policy to subdomains.
    - report-uri: URL to which validation failures are reported.

2. Benefits of HPKP

1. **Prevent Rogue Certificates**

- Ensures that only pre-specified public keys are accepted, even if a malicious or compromised CA issues a certificate for the domain.

2. **Protection Against MITM Attacks**

- MITM attackers cannot forge certificates unless they have access to the pinned private keys.

3. **Enhanced Security for HTTPS**

- Provides an additional layer of trust beyond the CA system.

3. Risks and Challenges of HPKP

1. **Risk of Permanent Lockout**

- If the pinned keys are lost (e.g., due to key rotation or server mismanagement), the domain becomes inaccessible for all users until the pins expire.
- Example: A misconfigured header with invalid pins could render a website completely unusable.

2. **Complexity**

- HPKP required careful key management and backup strategies to avoid accidental lockouts.
- Many administrators found it challenging to implement correctly.

3. **Abuse Potential**

- Attackers who gained temporary control of a server could set malicious pins, effectively hijacking or bricking the domain for users.

4. **Reporting Issues**

- The report-uri directive was often underutilized, making it difficult for site owners to detect and address pinning failures.

4. Deprecation of HPKP

Why HPKP Was Deprecated

- Adoption Challenges
  - HPKP adoption was low due to its complexity and high risk of misconfiguration.
- Security Concerns
  - Misuse and abuse of HPKP posed a greater risk than the attacks it aimed to mitigate.
- Better Alternatives
  - Mechanisms like **Certificate Transparency (CT) and HSTS (HTTP Strict Transport Security)** were deemed more effective and safer.

Timeline of Deprecation

- Google Chrome deprecated HPKP in version 67 (May 2018).
- Other browsers followed suit, effectively making HPKP obsolete.

5. Alternatives to HPKP

1. **Certificate Transparency (CT)**

- Provides a publicly auditable log of issued certificates.
- Allows domain owners to monitor for unauthorized certificates.

2. **HTTP Strict Transport Security (HSTS)**

- Enforces HTTPS connections, ensuring secure communication.

- Combined with Certificate Transparency, HSTS reduces the likelihood of MITM attacks.

3. **DNS-Based Authentication of Named Entities (DANE)**

- Relies on DNSSEC to bind certificates to domain names securely.

4. **Expect-CT Header**

- Encourages the use of Certificate Transparency by requiring certificates to be logged.

6. Example of Deprecated HPKP Implementation

Legacy Header (Not Recommended)

```
Public-Key-Pins:
    pin-sha256="AbCdEfGhIjKlMnOpQrStUvWxYz1234567890abcDEF=";
    pin-sha256="XyZAbC123EfGhIjKlMnOpQrStUvWxYz456789abcDEF=";
    max-age=5184000;
    includeSubDomains;
    report-uri="https://example.com/report";
```

Current Best Practices

- Use HSTS:

```
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
```

- Enable Certificate Transparency:
  - Ensure certificates are logged in CT during issuance.

7. Summary

Aspect	Details
What is HPKP?	A mechanism to pin public keys for a domain to prevent rogue certificates.
How It Worked	Used Public-Key-Pins header to specify allowed keys and durations.
Key Benefits	Protected against rogue certificates and MITM attacks.
Challenges	Risk of lockout, complexity, and potential for abuse.
Deprecation	Deprecated in Chrome 67 (2018) due to risks and low adoption.
Alternatives	Certificate Transparency (CT), HSTS, and Expect-CT.

While HPKP was a promising concept for enhancing HTTPS security, **its risks and complexities outweighed its benefits**. Modern web security practices now rely on safer and more robust mechanisms like Certificate Transparency and HSTS, which provide equivalent or superior protection with less risk to website administrators.