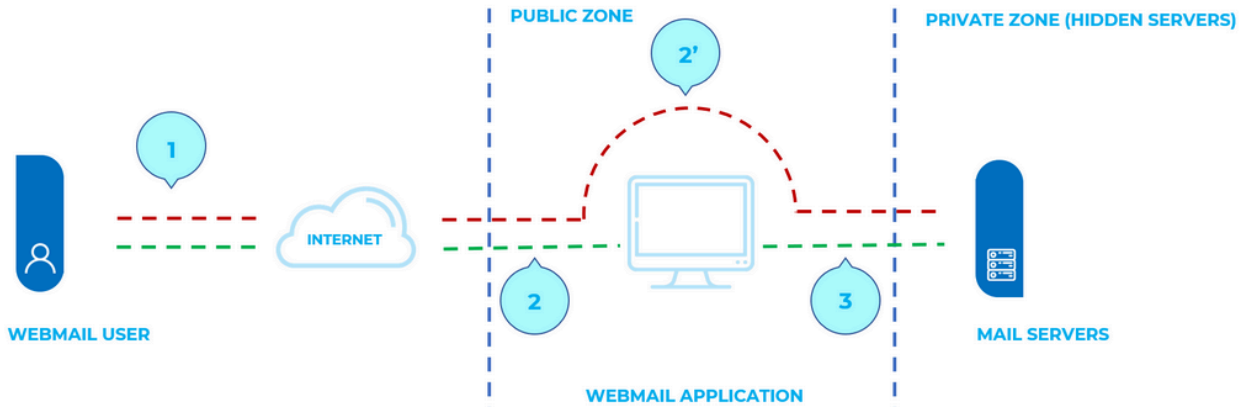
Testing for IMAP SMTP Injection (IMAP SMTP Enjeksiyonu için Test)

Summary (Özet)

Bu tehdit, posta sunucuları (IMAP / SMTP) ile iletişim kuran tüm uygulamaları etkiler, genellikle web postası uygulamaları. Bu testin amacı, veri düzgün bir şekilde sterilize edilmemesi nedeniyle, posta sunucularına keyfi IMAP / SMTP komutlarını enjekte etme kapasitesini doğrulamaktır.

IMAP / SMTP Enjeksiyon tekniği, posta sunucusuna doğrudan İnternet'ten erişilemediği takdirde daha etkilidir. Arka uç posta sunucusuyla tam iletişimin mümkün olduğu durumlarda, doğrudan test yapılması önerilir.

Bir IMAP / SMTP Enjeksiyonu, aksi takdirde İnternet'ten doğrudan erişilemeyecek bir posta sunucusuna erişmeyi mümkün kılar. Bazı durumlarda, bu dahili sistemler, ön uç web sunucularına uygulanan aynı düzeyde altyapı güvenliğine ve sertleştirmeye sahip değildir. Bu nedenle, posta sunucusu sonuçları son kullanıcılar tarafından yapılan saldırılara karşı daha savunmasız olabilir (B.Ünistçe 1'de sunulan şemaya bakın.



Şekil 4.7.10-1: IMAP / SMPT Enjeksiyon tekniğini kullanarak posta sunucularıyla iletişim

Şekil 1, web posta teknolojilerini kullanırken genellikle görülen trafik akışını tasvir eder. Adım 1 ve 2, web posta istemcisi ile etkileşime giren kullanıcıyken, 2. adım, web posta istemcisini atlayan ve doğrudan arka uç posta sunucularıyla etkileşime giren test cihazıdır.

Bu teknik çok çeşitli eylemlere ve saldırılara izin verir. Olasılıklar enjeksiyonun türüne ve kapsamına ve test edilen posta sunucusu teknolojisine bağlıdır.

IMAP / SMPT Enjeksiyon tekniğini kullanan bazı saldırı örnekleri şunlardır:

- IMAP/SMTP protokolündeki güvenlik açıklarının sömürülmesi
- Uygulama kısıtlamaları kaçağı
- Otomatlanmayı önleme süreci kaçakçılığı
- Bilgi sızıntıları
- Röle / SPAM

Test Objectives (Test Hedefleri)

- IMAP/SMTP enjeksiyon noktalarını belirleyin.
- Sistemin veri akışını ve dağıtım yapısını anlayın.
- Enjeksiyon etkilerini değerlendirin.

How To Test (Nasıl Test Edilir)

Identifying Vulnerable Parameters (Savunmasız Parametreleri Tanımlamak)

Savunmacı, savunmasız parametreleri tespit etmek için, uygulamanın girişle ilgili yeteneğini analiz etmelidir. Giriş doğrulama testi, test cihazının sunucuya sahte veya kötü amaçlı istek göndermesini ve yanıtı analiz etmesini gerektirir. Güvenli bir uygulamada, yanıt, müşteriye bir şeylerin yanlış gittiğini söyleyen bazı ilgili eylemlerle ilgili bir hata olmalıdır. Savunmasız bir uygulamada, kötü amaçlı istek bir ile cevap verecek arka uç uygulaması tarafından işlenebilir `HTTP 200 OK` Cevap mesajı. Gönderilen taleplerin test edilen teknolojiyle eşleşmesi gerektiğini belirtmek önemlidir. Bir MySQL sunucusu kullanıldığında Microsoft SQL sunucusu için SQL enjeksiyon dizeleri göndermek yanlış olumlu yanıtlarla sonuçlanacaktır.

Bu durumda, kötü niyetli IMAP komutları göndermek, IMAP test edilen altta yatan protokol olduğundan, modus operandi'dir.

Kullanılması gereken IMAP özel parametreler şunlardır:

On the IMAP server	On the SMTP server
Authentication	Emissor email
operations with mail boxes (list, read, create, delete, rename)	Destination email
operations with messages (read, copy, move, delete)	Subject
Disconnection	Message body
	Attached files

IMAP sunucusunda	SMTP sunucusunda
Kimlik doğrulaması	Emissör e-postası
Posta kutuları ile işlemler (listeleyin, okuyun, oluşturun, silme, yeniden adlandırın)	Destinasyon e-postası
Mesajlarla çalışır (oku, kopyalayın, hareket edin, silin)	Konu
Bağlantı Kesimi	Mesaj gövdesi
	Ekli dosyalar

Bu örnekte, “posta kutusu” parametresi, tüm talepleri parametre ile manipüle edilerek test ediliyor:

`http://<webmail>/src/read_body.php?mailbox=INBOX&passed_id=46106&startMessage=1`

Aşağıdaki örnekler kullanılabilir.

- Parametreye bir fındık değeri atayın:

`http://<webmail>/src/read_body.php?mailbox=&passed_id=46106&startMessage=1`

- Değeri rastgele bir değerle yerine getirin:

`http://<webmail>/src/read_body.php?mailbox=NOTEXIST&passed_id=46106&startMessage=1`

- Parametreye başka değerler ekleyin:

`http://<webmail>/src/read_body.php?mailbox=INBOX PARAMETER2&passed_id=46106&startMessage=1`

- Standart olmayan özel karakterler ekleyin (yani: `\`, `'`, `"`, `@`, `#`, `!`, `|`) ::

`http://<webmail>/src/read_body.php?mailbox=INBOX"&passed_id=46106&startMessage=1`

- Parametreyi ortadan kaldırın:

`http://<webmail>/src/read_body.php?passed_id=46106&startMessage=1`

Yukarıdaki testin nihai sonucu, testçiye üç olası durum sağlar:
S1 - Uygulama bir hata kodu / mesaj döndürür
S2 - Uygulama bir hata kodu/mesajı iade etmez, ancak istenen işlemi gerçekleştirmez
S3 - Uygulama bir hata kodu / mesaj iade etmez ve istenen işlemi normal şekilde gerçekleştirir

Durumlar S1 ve S2 başarılı IMAP / SMTP enjeksiyonunu temsil eder.

Bir saldırganın amacı, uygulamanın enjeksiyon ve daha fazla manipülasyona karşı savunmasız olduğunun bir göstergesi olduğu için S1 yanıtını almaktır.

Bir kullanıcının aşağıdaki HTTP isteğini kullanarak e-posta başlıklarını aldığını varsayalım:

`http://<webmail>/src/view_header.php?mailbox=INBOX&passed_id=46105&passed_ent_id=0`

Bir saldırgan, karakteri enjekte ederek parametre INBOX'un değerini değiştirebilir `"` (%22 URL kodlamayı kullanarak):

`http://<webmail>/src/view_header.php?mailbox=INBOX%22&passed_id=46105&passed_ent_id=0`

Bu durumda, başvuru cevabı şu olabilir:

```
ERROR: Bad or malformed request.
Query: SELECT "INBOX""
Server responded: Unexpected extra arguments to Select
```

S2'nin başarılı bir şekilde test edilmesi daha zordur. Test cihazının, sunucunun savunmasız olup olmadığını belirlemek için kör komut enjeksiyonu kullanması gerekir.

Öte yandan, son durum (S3) bu paragrafta çürütülmüyor.

> ## Savunmasız parametreler listesi
>
> - Etkilenen işlevsellik

> - Olası enjeksiyon türü (IMAP / SMTP)

Understanding the Data Flow and Deployment Structure of the Client (Müşterinin Veri Akışı ve Dağıtım Yapısını Anlamak)

Tüm savunmasız parametreleri belirledikten sonra (örneğin, `passed_id` Testçinin enjeksiyonun ne kadar mümkün olduğunu belirlemeli ve daha sonra uygulamayı daha fazla kullanmak için bir test planı tasarlamalıdır.

Bu test durumunda, uygulamanın olduğunu tespit ettik `passed_id` Parametre savunmasızdır ve aşağıdaki talepte kullanılır:

`http://<webmail>/src/read_body.php?mailbox=INBOX&passed_id=46225&startMessage=1`

Aşağıdaki test vakasını kullanarak (namerik bir değer gerektiğinde alfabetik bir değer sağlamak):

`http://<webmail>/src/read_body.php?mailbox=INBOX&passed_id=test&startMessage=1`

Aşağıdaki hata mesajını oluşturacaktır:

```
ERROR : Bad or malformed request.
Query: FETCH test:test BODY[HEADER]
Server responded: Error in IMAP command received by server.
```

Bu örnekte, hata mesajı, yürütülen komutun adını ve ilgili parametreleri iade etti.

Diğer durumlarda, hata mesajı (`not controlled` Uygulama ile) İdam edilen komutun adını içerir, ancak uygun RFC'yi okumak, test cihazının diğer olası komutların neler yapılabileceğini anlamasını sağlar.

Uygulama tanımlayıcı hata mesajlarını iade etmezse, test cihazının yukarıda belirtilen işlevsellikle ilişkili tüm olası komutları (ve parametreleri) çıkarmak için etkilenen işlevselliği analiz etmesi gerekir. Örneğin, e-posta kutusu işlevselliğinde savunmasız bir parametre tespit edildiyse, etkilenen IMAP komutunun olduğunu varsaymak mantıklıdır. `CREATE` . . RFC'ye göre, `CREATE` Komut, oluşturmak için posta kutusunun adını belirten bir parametreyi kabul eder.

> Etkilenen IMAP / SMTP komutları listesi

> - Etkilenen IMAP / SMP komutlarından beklenen parametrelerin türü, değeri ve sayısı

IMAP/SMTP Command Injection (IMAP / SMTP Komuta Enjeksiyonu)

Test cihazı savunmasız parametreleri tanımladıktan ve yürütüldükleri bağlamı analiz ettikten sonra, bir sonraki aşama işlevsellikten yararlanıyor.

Bu aşamanın iki olası sonucu vardır:

1. Enjeksiyon, doğrulanmamış bir durumda mümkündür: etkilenen işlevsellik, kullanıcının doğrulanmasını gerektirmez. Mevcut enjekte edilen (IMAP) komutlar aşağıdakilerle sınırlıdır: `CAPABILITY`, `NOOP`, `AUTHENTICATE`, `LOGIN`, ve `LOGOUT`. .
2. Enjeksiyon sadece doğrulanmış bir durumda mümkündür: başarılı sömürü, test devam etmeden önce kullanıcının tam olarak doğrulanmasını gerektirir.

Her durumda, bir IMAP / SMTP Enjeksiyonunun tipik yapısı aşağıdaki gibidir:

- Başlık: beklenen komutun sonu;
- Vücut: yeni komutun enjeksiyonu;
- Footer: Beklenen komutun başlangıcı.

Bir IMAP / SMPP komutunu yerine getirmek için önceki komutun CRLF ile sonlandırılması gerektiğini hatırlamak önemlidir. `%0d%0a`) sıra.

Diyelim ki, tanımlayıcı savunmasız parametreler aşamasında, saldırganın parametreyi tespit ettiğini varsayalım. `message_id` Aşağıdaki istekte savunmasızdır:

`http://<webmail>/read_email.php?message_id=4791`

Ayrıca, 2. aşamada gerçekleştirilen analizin sonucunun (“Müveterinin veri akışını ve dağıtım yapısını anlamak”), bu parametreyle ilişkili komutu ve argümanları aşağıdaki şekilde tanımlamıştır:

`FETCH 4791 BODY[HEADER]`

Bu senaryoda, IMAP enjeksiyon yapısı şunlar olacaktır:

`http://<webmail>/read_email.php?message_id=4791 BODY[HEADER]%0d%0aV100 CAPABILITY%0d%0aV101 FETCH 4791`

Bu da aşağıdaki komutları oluşturur:

```
???? FETCH 4791 BODY[HEADER]
V100 CAPABILITY
V101 FETCH 4791 BODY[HEADER]
```

Nerede:

```
Header = 4791 BODY[HEADER]
Body   = %0d%0aV100 CAPABILITY%0d%0a
Footer = V101 FETCH 4791
```

> Etkilenen IMAP / SMTP komutları listesi
>
> - Keyfi IMAP / SMPP komut enjeksiyonu

Referances (Referanslar)

WhitePapers (Beyaz kağıtlar)

- RFC 0821 “Simple Mail Transfer Protocol”
- RFC 3501 “Internet Message Access Protocol - Version 4rev1”
- Vicente Aguilera Díaz: “MX Injection: Capturing and Exploiting Hidden Mail Servers”