CAPEC Nedir ve Kullanımı

a. CAPEC Nedir?

CAPEC, CWE ile kategorize edilmiş açıklık türleri listesindeki açıklıkların hangi tür saldırılarla sömürülebileceğini sunan bir saldırı deseni numaralandırma ve sınıflandırma standardıdır.

Saldırı deseni (attack pattern) demek saldırganlarca bir açıklığı sömürmede kullanılan yaklaşımı ifade eder. Saldırı desenlerinden bazısı şu şekildedir:

- Http Response Splitting (CAPEC-34)
- Session Fixation (CAPEC-61)
- Cross Site Request Forgery (CAPEC-62)
- SQL Injection (CAPEC-66)
- Cross-Site Scripting (CAPEC-63)
- Buffer Overflow (CAPEC-100)
- Clickjacking (CAPEC-103)
- Relative Path Traversal (CAPEC-139)
- ...

CAPEC standardı yukarıdakilerin de yer aldığı saldırı deseni listesi sunar. Bu listedeki saldırı desenlerinin her birinde açıklığın nasıl keşfedilebileceği, açıklığın sömürülüp sömürülemeyeceğinin nasıl anlaşılabileceği, ve açıklığın nasıl sömürülebileceği gibi detaylı bilgiler yer alır. Yani saldırı deseni listesinde saldırıların dökümantasyonu sunulur ve kötü niyetli kimselerin ilgili saldırıda hangi adımları takip ettiği öğrenilir. Bu şekilde siber dünyadaki ürün geliştiricilerinin yapılan saldırıları ve unsurlarını daha iyi anlaması ve bu şekilde ürünlerine doğru önlemler alabilmesi hedeflenir. CAPEC ürün geliştiricileri için güvenlik anlamında eğitim amaçlı oluşturulmuş bir standarttır.

CAPEC'de saldırı deseni listesindeki bir saldırı ve CWE'de açıklık türü listesindeki bir açıklık arasında one to one (bire bir) ilişki olmak zorunda değildir. Örneğin CAPEC'deki bir saldırı türü CWE'deki birden fazla açıklık türünü sömürebilir. Dolayısıyla CAPEC'in CWE ile her zaman bire bir ilişkisi olmayabilir. Bazen saldırı deseni ve açıklık türü arasında one to n (bire n) ilişki söz konusu olmaktadır (bkz. b. Capec Nasıl Kullanılır?#xiv) "Related Weaknesses" Bölümü Başlığı).

Aşağıda CAPEC, CWE ve CVE için işlevlerini açıklayan bir şema verilmiştir.

Yukarıdaki şemadan görülebileceği gibi CAPEC CWE'deki açıklıkların nasıl sömürülebileceğini gösteren bir standarttır.

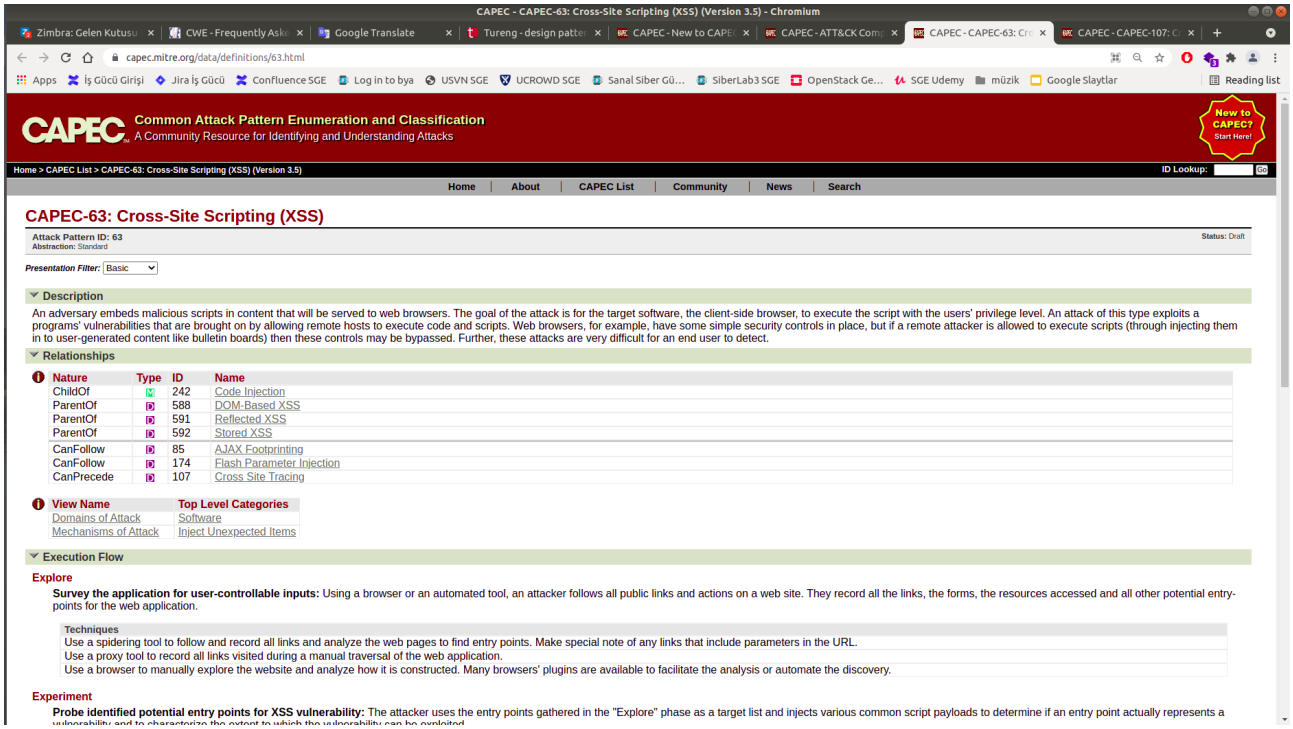
b. Capec Nasıl Kullanılır?

CAPEC'de saldırı deseni dökümantasyon sayfaları kullanımı şu şekildedir:

i) Bir Capec Saldırı Deseni Sayfası

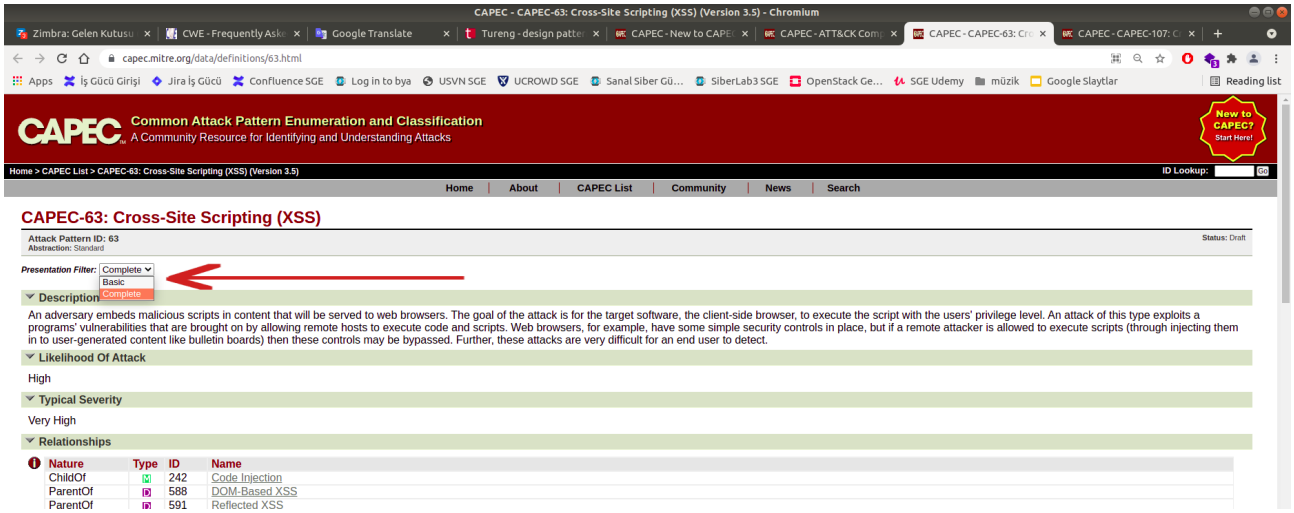
Örnek bir saldırı deseni sayfasına gidelim. Örn; CAPEC-63: Cross-Site Scripting

https://capec.mitre.org/data/definitions/63.html



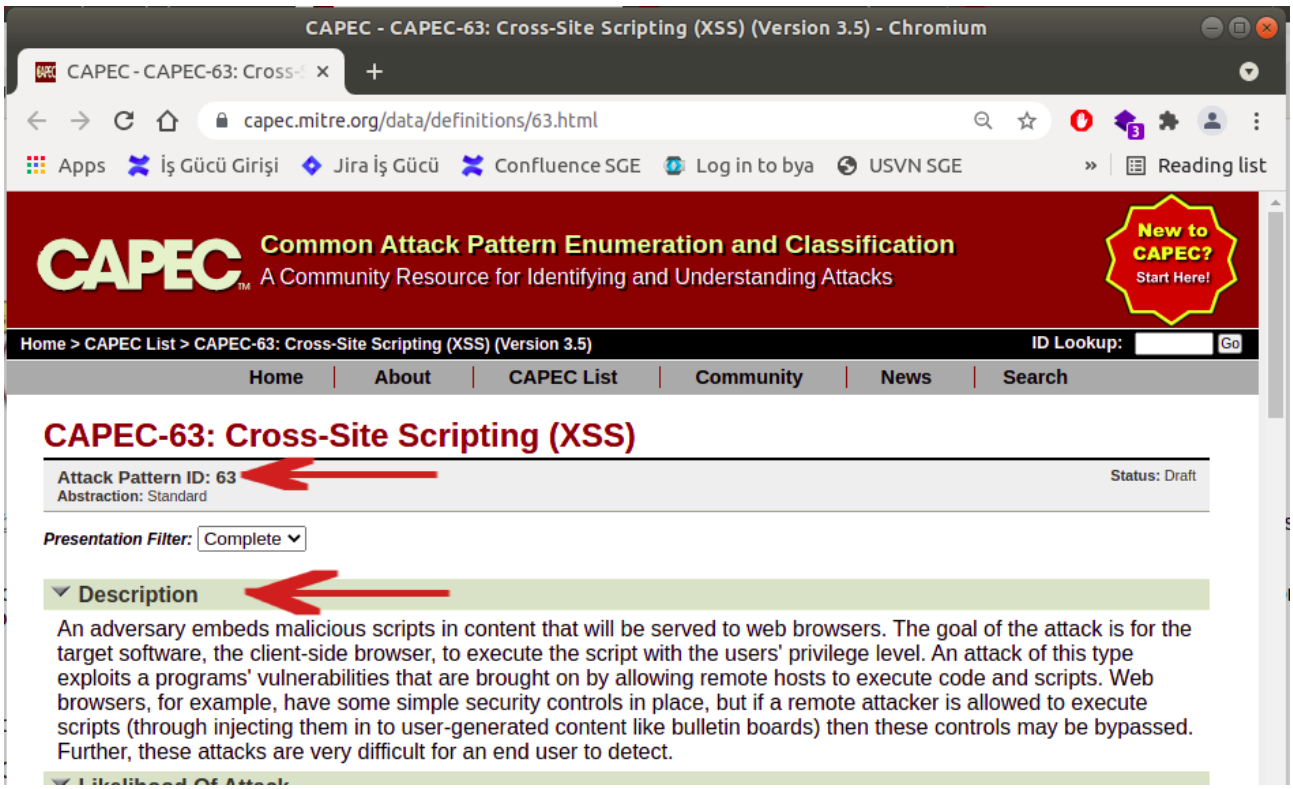
ii) "Presentation Filter" Filtrelemesi

Saldırı deseni sayfalarında saldırı deseninin tüm detaylarını görebilmek adına sol üst köşedeki "Presentation Filter (Sunum Filtrelemesi) seçeneği Basic'den Complete'e çekilmelidir.



iii) “Attack Pattern ID” ve “Description” Bölümü

Saldırı deseni sayfalarında ilk başta saldırı desenlerine verilen numara “Attack Pattern ID (Saldırı Deseni ID)” ile gösterilir. Ardından saldırı desenlerinin ne hakkında olduğu “Description (Tanımlama)” bölümünde açıklanır.



Örneğin CAPEC-63: Cross Site Scripting saldırı deseni sayfasında “Attack Pattern ID” 63 belirtildiğinden CAPEC numaralandırması CAPEC-63 şeklinde olur.

iv) “Likelihood of Attack” Bölümü

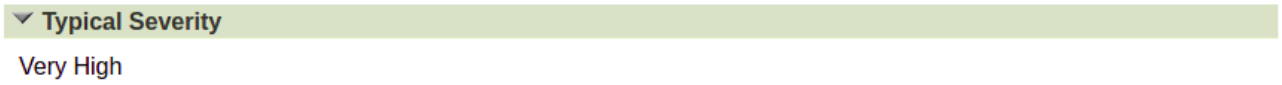
Saldırı deseni sayfalarında daha sonra “Likelihood of Attack (Saldırı Olasılığı)” bölümü yer alır. Bu bölümde CAPEC yetkililerinin yaptıkları kişisel bir analiz doğrultusunda piyasadaki ürünlerin ilgili saldırı deseniyle belirlenen bir periyot süresince ne kadar sıklıkta karşılaşabilecekleri bilgisi yer alır.

Örneğin Capec 63: Cross-Site Scripting saldırı deseni sayfasında “Likelihood of Attack” bölümünde High dendiği için piyasadaki ürünlerin bu saldırı deseniyle karşılaşma sıklığı çok fazladır denmektedir. Örneğin “Likelihood of Attack” bölümünde Low denseydi piyasadaki ürünlerin bu saldırı deseniyle karşılaşma sıklılığı nadirdir denebilirdi.

Sonuç olarak Likelihood of Attack bölümü piyasadaki ürünlerin ilgili saldırıyla karşılaşma sıklığını belirtir. Diğer bir ifadeyle ilgili saldırıya maruz kalma sıklığını belirtir. Son bir cümle olarak ilgili saldırının gerçekleşme sıklığını belirtir.

v) “Typical Severity” Bölümü

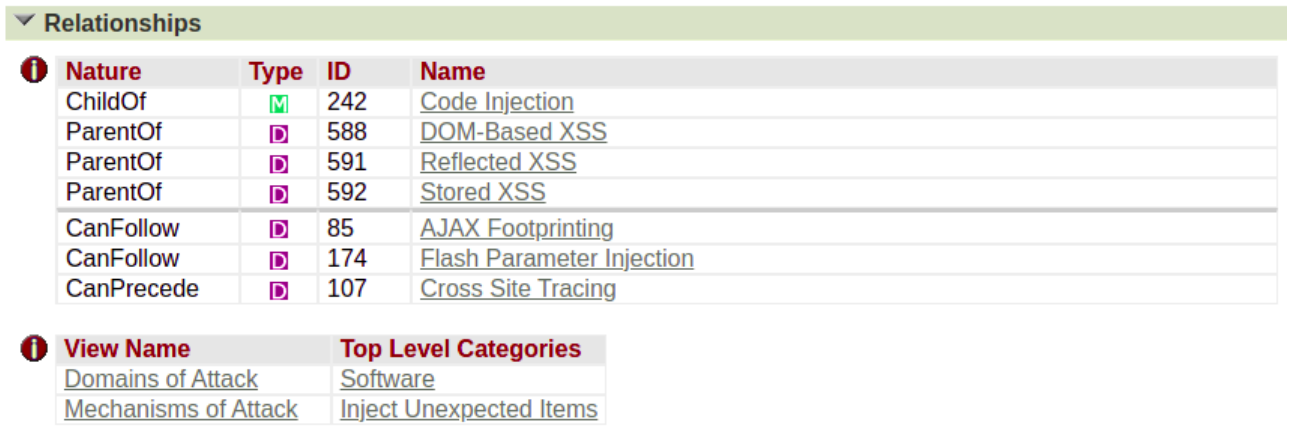
Saldırı deseni sayfalarında daha sonra “Typical Severity (Tipik Önem Derecesi)” bölümü yer alır. Bu bölümde saldırıların verebileceği zarara göre saldırıların ne kadar kritik bir saldırı olduğu seviyelendirmesi yapılır.



Örneğin CAPEC-63: Cross-Site Scripting saldırı türünün kritikliğine “Very High” denerek çok ciddi zarar veren bir saldırı türü olduğu belirtilmiştir.

vi) “Relationship” Bölümü

Saldırı deseni sayfalarında daha sonra “Relationship (İlişkiler)” bölümü yer alır. Bu bölümde CAPEC veritabanında yer alan tüm saldırı desenlerinin sınıflı bir şekilde dizildiği bir ağaçta saldırı deseninin diğer saldırı desenlerine göre hangi hiyerarşik konumda yer aldığı gösterilir.



Örneğin CAPEC-63: Cross-Site Scripting saldırı deseni sayfasında Nature tablosunda bu saldırı deseninin tüm saldırı desenleri ağacında hangi saldırı deseninin chid'ı olduğu, sonra hangi saldırı deseninin parent'ı olduğu ve sonra hangi saldırı desenleriyle komşu olduğu (yani ağaçtaki düğümlerde bir öncesindekiler ve bir sonrasındakiler) gösterilmektedir.

Bundan hareketle Nature tablosunda ilk satırda belirtildiği gibi Capec-63: Cross-Site Scripting saldırı deseni Code Injection saldırı deseninin child'ıymış. Yani cross-site scriptig saldırı deseni bir code injection türüymüş.

Nature tablosunda ikinci, üçüncü ve dördüncü satırlarda belirtildiği gibi CAPEC-63: Cross-Site Scripting saldırı deseni DOM-Based XSS, Reflected XSS ve Stored XSS'in parent'ıymış. Yani cross-site script saldırı deseni alt türlere sahipmiş.

Nature tablosunda beşinci, altıncı ve yedinci satırlarda belirtildiği gibi CAPEC-63: Cross-Site Scripting saldırı deseni AJAX Footprinting ve Flash Parameter Injection saldırı desenlerinden hiyerarşik olarak ağaçta aynı yükseklikte fakat onlardan sonra gelmekteymiş ve Cross Site Tracing saldırı deseninden hiyerarşik olarak ağaçta aynı yükseklikte, fakat ondan bir önce gelmekteymiş.

Sonuç olarak CAPEC'de veritabanında yüklü tüm saldırı desenleri bir ağaca konulur ve kökten dallanarak ilerleyen ağaçta her bir saldırı deseni ilişkili halde sınıflı dizilir. CAPEC'de tüm saldırı desenleri bu şekilde kategorize edilmişlerdir.

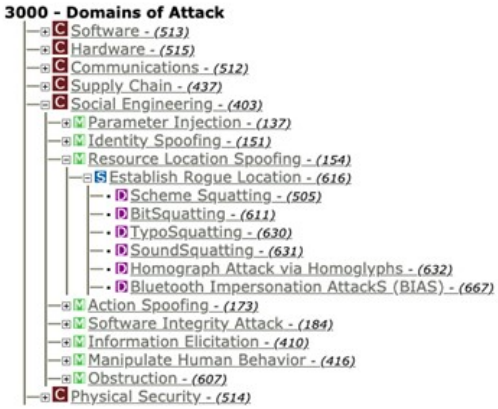
> Not:
>
> CWE açıklık türü sayfalarında da aynı şey söz konusudur. CWE açıklık türü sayfalarında her bir açıklığın Relationship bölümünde parent, child ve follow, precede ilişkileri sunulur. Dolayısıyla CWE açıklıkları kökten ilerleyerek dallanan bir ağaçta hiyerarşik ve ilişkili olarak sınıflandırılmışlardır. Bu şekilde tüm cwe açıklık türleri veritabanında kategorize edilmişlerdir.

Relationship bölümünde View Name tablosunda ise CAPEC'de veritabanında yüklü tüm saldırı desenleri için iki farklı kök kategori sunulur. Bu kök kategoriler Domains of Attack (Türkçesiyle; Saldırı Alanları) ve Mechanics of Attack (Türkçesiyle; Saldırı Mekanikleri) şeklindedir. Bu iki kök kategori CAPEC veritabanındaki tüm saldırı desenlerini iki ağaçta iki farklı bakış açısıyla sınıflar. Domains of Attack kök kategorisi bir ağaçta veritabanındaki tüm saldırı desenlerini kapsamına girdikleri saldırı alanlarına göre sınıflar. Mechanics of Attack kök kategorisi bir ağaçta veritabanındaki tüm saldırı desenlerini kapsamına girdikleri saldırı yöntemlerine göre sınıflar.

Relationship bölümü View Name tablosunda yer alan Domains of Attack (Saldırı Alanları) satırında ilgili saldırı deseninin "saldırı alanlarına göre sınıflandırılmış" tüm saldırı desenleri ağacında ait olduğu en üst seviye kategori bilgisi paylaşılır, View Name tablosunda yer alan Mechanics of Attack (Saldırı Mekanikleri) satırında ise ilgili saldırı deseninin "saldırı yöntemlerine göre sınıflandırılmış" tüm saldırı desenleri ağacında ait olduğu en üst seviye kategori bilgisi paylaşılır.

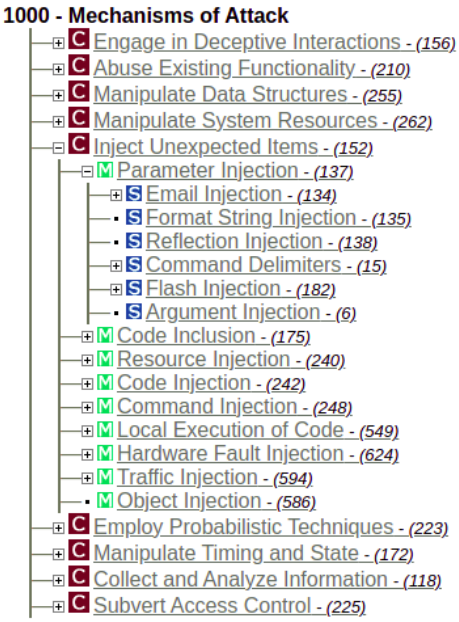
Bir saldırı deseni sayfasında bu iki farklı kategorik sınıflandırma ile View Name tablosundaki Domains of Attack kök kategorisi satırından saldırı deseniyle aynı saldırı alanında kullanılan başka saldırı desenleri keşfedilebilir veya View Name tablosundaki Mechanics of Attack kök kategorisi satırından saldırı deseniyle aynı saldırı yöntemini kullanan başka saldırı desenleri keşfedilebilir.

Domains of Attack (Saldırı Alanları) kök kategorisine göz atacak olursak Yazılım, Donanım, …, Sosyal Mühendislik, Fiziksel Güvenlik gibi en üst seviye (top level) kategoriler (saldırı alanları) yer almaktadır.



Bu en üst seviye kategoriler altında saldırı desenleri kategorik olarak yer almaktadır.

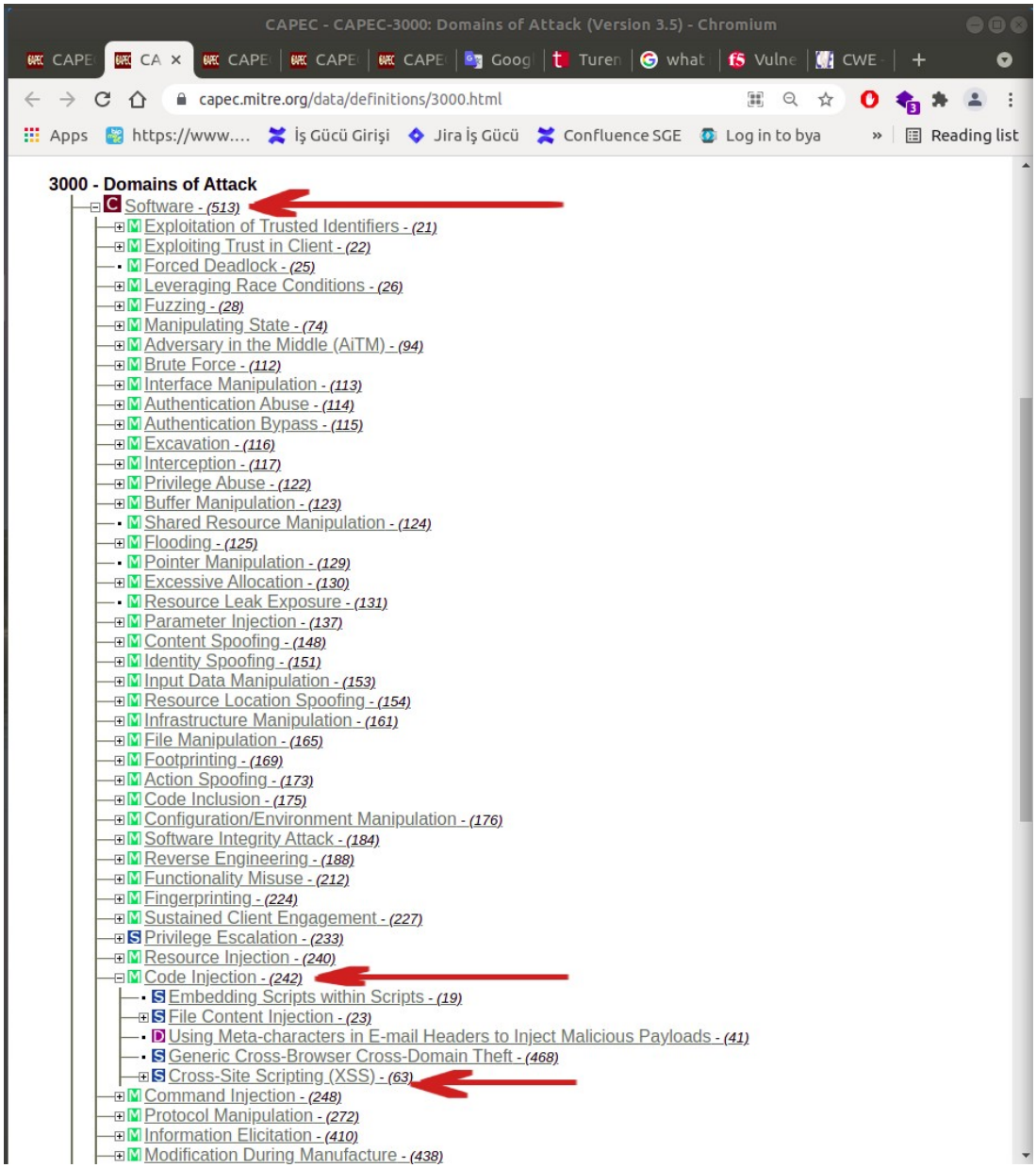
Mechanics of Attack (Saldırı Mekanikleri) kök kategorisine göz atacak olursak Aldatıcı Etkileşimlerde Bulunma, Var Olan Fonksiyonelliği Suistimal Etme, …, Erişim Kontrolünü Bozma gibi en üst seviye (top level) kategoriler (saldırı yöntemleri) yer almaktadır.



Bu en üst seviye kategoriler altında saldırı desenleri kategorik olarak yer almaktadır.

Örneğin CAPEC-63: Cross-Site Scripting saldırı deseni sayfasında View Name tablosunda saldırı deseninin Domains of Attack (Saldırı Alanları) kök kategorisinin içerisindeki Software (Yazılım) en üst seviye kategorisinde yer aldığı belirtilmiştir. Belirtildiği gibi Software kategorisinden aşağılara elle gidildiğinde Cross-Site Scripting saldırı deseni görülebilecektir.

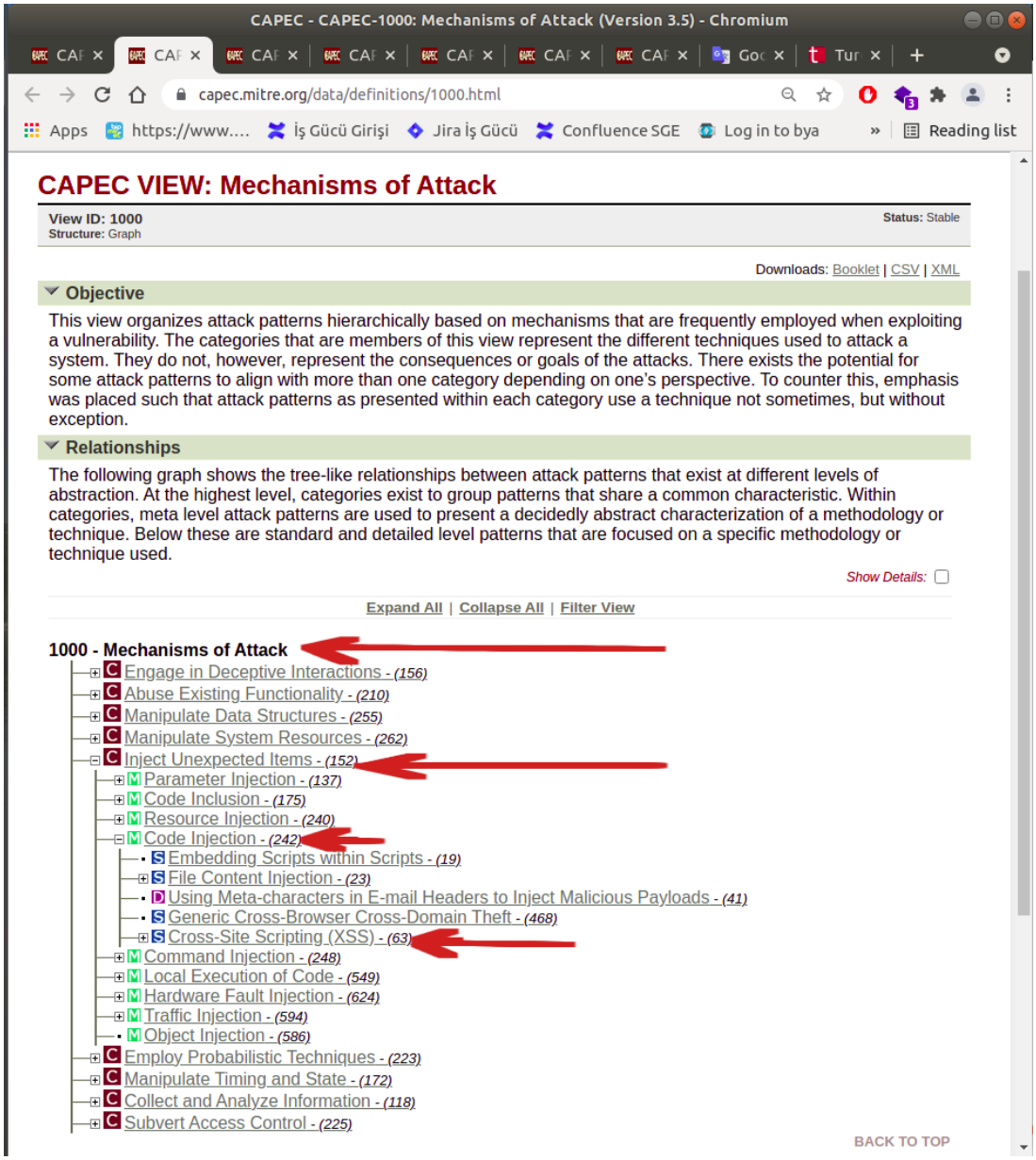
Domains of Attack -> “Software” -> Code Injection -> Cross-Site Scripting (XSS) (63).



Bunun gibi Domains of Attack (Saldırı Alanları) ağacında tüm saldırı desenleri Yazılım alanına ait bir saldırı deseniyse Yazılım en üst seviye kategorisi altında, Donanım alanına ait bir saldırı deseniyse Donanım en üst seviye kategorisi altında,…, Fiziksel Güvenlik alanına ait bir saldırı deseniyse Fiziksel Güvenlik en üst seviye kategorisi altında kategorik olarak yer alırlar. Domains of Attack kök kategorisinde veritabanındaki tüm saldırı desenleri ait oldukları saldırı alanına göre sınıflı dizilidirler.

Örneğin CAPEC-63: Cross-Site Scripting saldırı deseni sayfasında View Name tablosunda saldırı deseninin Mechanics of Attack (Saldırı Mekanikleri) kök kategorisinin içerisindeki Inject Unexpected Items (Umulmayan Öğeler Enjekte Etme) en üst seviye kategorisinde yer aldığı belirtilmiştir. Belirtildiği gibi Inject Unexpected Items kategorisinden aşağılara elle gidildiğinde Cross-Site Scripting saldırı deseni görülebilecektir.

Mechanics of Attack -> “Inject Unexpected Items” -> Code Injection -> Cross-Site Scripting (XSS) (63).



Bunun gibi Mechanics of Attack (Saldırı Mekanikleri) ağacında tüm saldırı desenleri Aldatıcı Etkileşimlerde Bulunma yöntemine ait bir saldırı deseniyse Aldatıcı Etkileşimlerde Bulunma en üst seviye kategorisi altında, Var Olan Fonksiyonelliği Suistimal Etme yöntemine ait bir saldırı deseniyse Var Olan Fonksiyonelliği Suistimal Etme en üst seviye kategorisi altında,…, Erişim Kontrolünü Bozma alanına ait bir saldırı deseniyse Erişim Kontrolünü Bozma en üst seviye kategorisi altında kategorik olarak yer alırlar. Mechanics of Attack kök kategorisinde veritabanındaki tüm saldırı desenleri ait oldukları saldırı yöntemine göre sınıflı dizilidirler.

Sonuç olarak Relationship bölümünde View Name tablosundaki Domains of Attack (Saldırı Alanları) satırında ilgili saldırı deseninin “saldırı alanlarına göre sınıflandırılmış” tüm saldırı desenleri ağacındaki en üst seviye kategori bilgisi paylaşılır. View Name tablosundaki ikinci satır olan Mechanics of Attack (Saldırı Mekanikleri) satırında ise ilgili saldırı deseninin “saldırı yöntemlerine göre sınıflandırılmış” tüm saldırı desenleri ağacındaki en üst seviye kategori bilgisi

paylaşılır. Bu şekilde aynı saldırı alanını kullanan veya aynı saldırı yöntemini kullanan başka saldırılar keşfedilebilir.

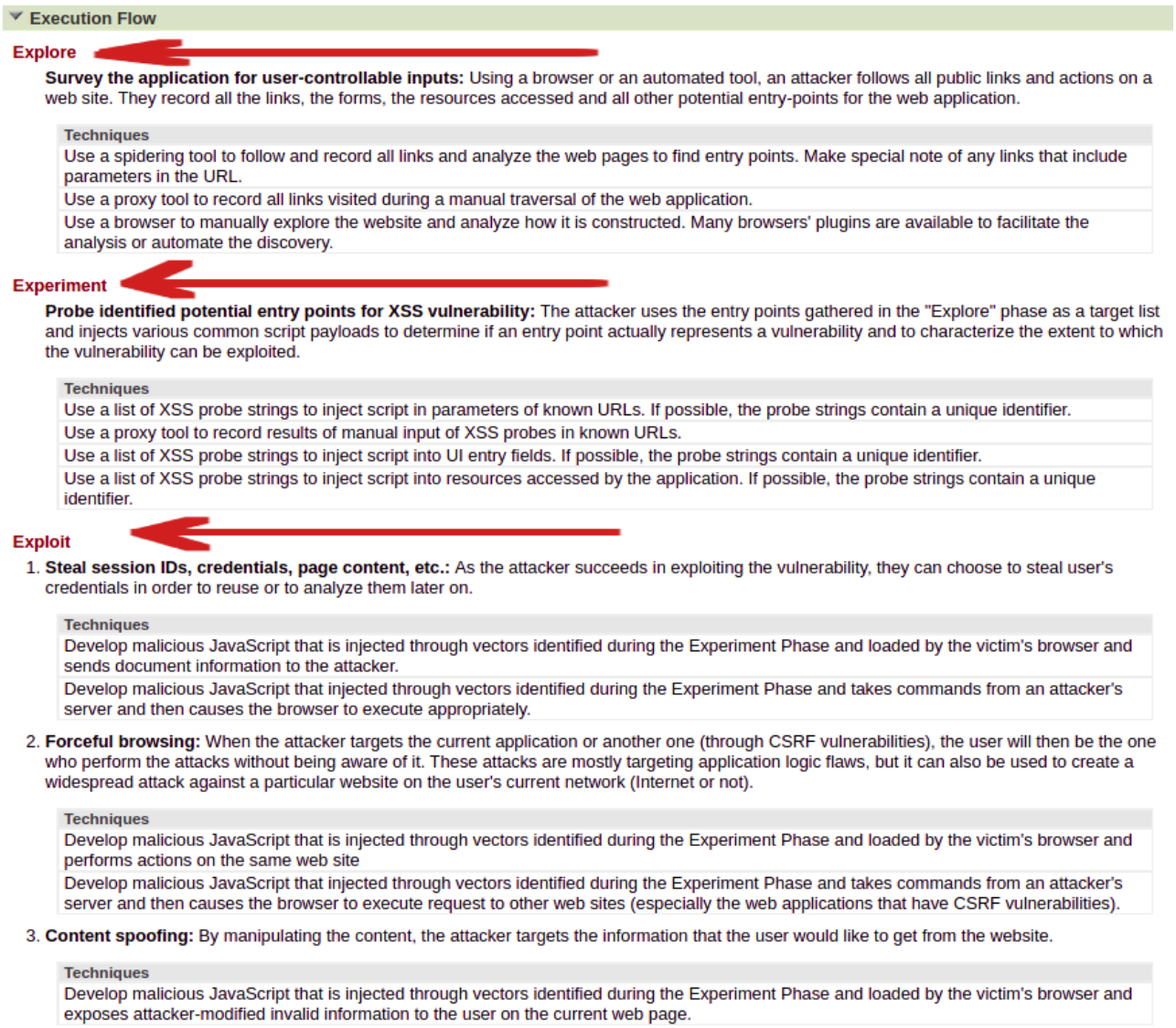
Not: Domain of Attacks (Saldırı Alanları) ve Mechanics of Attack (Saldırı Yöntemleri) kök kategorilerine View Name tablosundan tıklayarak veya şu linklerden gidilebilir;

https://capec.mitre.org/data/definitions/3000.html // Domain of Attacks
https://capec.mitre.org/data/definitions/1000.html // Mechanics of Attacks

Satırlar yavaşça yüklendikten sonra üst menüdeki Expand All linkine tıklanarak ve ilgili saldırı deseni ismi sayfada aratılarak aynı kategorideki başka saldırı desenleri öğrenilebilir.

vii) "Execution Flow" Bölümü

Saldırı deseni sayfalarında daha sonra "Execution Flow (Yürütme Akışı)" bölümü yer alır. Bu bölümde saldırı deseninin hedef tarafa uygulanması noktasında ilgili açıklığın nasıl keşfedilebileceği, ilgili açıklığın sömürülebilir olup olmadığının nasıl test edilebileceği ve sonra ilgili açıklığın nasıl sömürülebileceği teknik detay adımlarla yer alır.



Execution Flow

Explore

Survey the application for user-controllable inputs: Using a browser or an automated tool, an attacker follows all public links and actions on a web site. They record all the links, the forms, the resources accessed and all other potential entry-points for the web application.

Techniques
Use a spidering tool to follow and record all links and analyze the web pages to find entry points. Make special note of any links that include parameters in the URL.
Use a proxy tool to record all links visited during a manual traversal of the web application.
Use a browser to manually explore the website and analyze how it is constructed. Many browsers' plugins are available to facilitate the analysis or automate the discovery.

Experiment

Probe identified potential entry points for XSS vulnerability: The attacker uses the entry points gathered in the "Explore" phase as a target list and injects various common script payloads to determine if an entry point actually represents a vulnerability and to characterize the extent to which the vulnerability can be exploited.

Techniques
Use a list of XSS probe strings to inject script in parameters of known URLs. If possible, the probe strings contain a unique identifier.
Use a proxy tool to record results of manual input of XSS probes in known URLs.
Use a list of XSS probe strings to inject script into UI entry fields. If possible, the probe strings contain a unique identifier.
Use a list of XSS probe strings to inject script into resources accessed by the application. If possible, the probe strings contain a unique identifier.

Exploit

1. **Steal session IDs, credentials, page content, etc.:** As the attacker succeeds in exploiting the vulnerability, they can choose to steal user's credentials in order to reuse or to analyze them later on.

Techniques
Develop malicious JavaScript that is injected through vectors identified during the Experiment Phase and loaded by the victim's browser and sends document information to the attacker.
Develop malicious JavaScript that injected through vectors identified during the Experiment Phase and takes commands from an attacker's server and then causes the browser to execute appropriately.

2. **Forceful browsing:** When the attacker targets the current application or another one (through CSRF vulnerabilities), the user will then be the one who perform the attacks without being aware of it. These attacks are mostly targeting application logic flaws, but it can also be used to create a widespread attack against a particular website on the user's current network (Internet or not).

Techniques
Develop malicious JavaScript that is injected through vectors identified during the Experiment Phase and loaded by the victim's browser and performs actions on the same web site
Develop malicious JavaScript that injected through vectors identified during the Experiment Phase and takes commands from an attacker's server and then causes the browser to execute request to other web sites (especially the web applications that have CSRF vulnerabilities).

3. **Content spoofing:** By manipulating the content, the attacker targets the information that the user would like to get from the website.

Techniques
Develop malicious JavaScript that is injected through vectors identified during the Experiment Phase and loaded by the victim's browser and exposes attacker-modified invalid information to the user on the current web page.

Cross-Site Scripting Saldırı Desenine Ait “Execution Flow” Detay Bilgi:

Örneğin CAPEC-63: Cross-Site Scripting saldırı deseni sayfasında Execution Flow bölümünde saldırganların saldırı desenini kullanırken izledikleri yol, yani yürütme akışı şu şekilde sunulmuştur:

Explore (Keşif) Aşaması:

Cross-Site Scripting saldırı deseni için Explore aşamasında kullanıcı kontrollü girdi noktalarını ara şeklinde yön tayin edilmektedir. Bunun için saldırganların bir web tarayıcısı ya da otomatize bir araç kullanarak web sitedeki tüm public linkleri ve eylemleri kayıt altına alacağı ifade edilmektedir. Bu şekilde saldırganların web uygulama için tüm linkleri, tüm formları, tüm erişilen kaynakları ve potansiyel diğer tüm girdi noktalarını kayıt altına alacakları ifade edilmektedir.

Keşif aşamasında kullanıcı kontrollü girdi noktalarını tespit etmek için saldırganların teknik olarak kullanabilecekleri yöntemlerden ilkinde; saldırganların web uygulamadaki tüm linkleri kayıt altına almak ve bu kayıt altındaki linklerde kullanıcı kontrollü girdi noktalarını tespit etmek için otomatize bir Spidering aracı kullanabilecekleri söylenmektedir.

Keşif aşamasında kullanıcı kontrollü girdi noktalarını tespit etmek için saldırganların teknik olarak kullanabilecekleri alternatif yöntemlerden ikincisinde web tarayıcıda web uygulamayı manuel olarak gezinebilecekleri ve sadece gidilen linkleri kayıt altına almak ve bu linklerde kullanıcı kontrollü girdi noktalarını tespit etmek için bir proxy aracı kullanabilecekleri söylenmektedir.

Keşif aşamasında kullanıcı kontrollü girdi noktalarını tespit etmek için saldırganların teknik olarak kullanabilecekleri alternatif yöntemlerden üçüncüsünde ise web uygulamayı manuel olarak arayüzden inceleyebilecekleri, web uygulamanın nasıl inşa edildiğini gözlemleyebilecekleri ve kullanıcı kontrollü girdi noktalarını tespit etmek için bir web tarayıcı aracı kullanabilecekleri söylenmektedir. İlaveten bu teknik yöntemdeki keşif aşamasını kolaylaştırmak veya otomatize hale getirmek için birçok mevcut web tarayıcı aracı plugin’leri kullanılabileceği söylenmektedir.

Sonuç olarak keşif aşamasında saldırganlar 3 farklı teknik yolla kullanıcı kontrollü girdi noktaları kapsamını belirleyebilirler. Birincisinde spidering tool’u ile otomatik crawling yapma sonucu teker teker sonuçları inceleyerek, ikincisinde proxy tool’u ile konfigure edilmiş web tarayıcıda manuel gezinme yaparak ziyaret edilen linklerin kayıt altına alınması sonucu (yani manual crawling sonucu) teker teker ziyaret edilen linklerdeki sonuçları inceleyerek, üçüncüsünde web tarayıcı aracı ile web uygulama içerisinde gezinerek - seçime bağlı olarak plugin kullanımıyla - arayüzden teker teker ziyaret edilen ekranlarda incelemede bulunarak şeklindedir. Bu alternatif teknik yollardan her biri ile belirli oranda kullanıcı kontrollü girdi noktaları kapsamı belirlenebilir.

Experiment (Deneme) Aşaması:

Experiment (yani deneme) aşamasında keşif aşamasında elde edilen kullanıcı kontrollü girdi noktaları kapsamını saldırı deseninin uygulanabilirliği noktasında

yokla şeklinde yön tayin edilmektedir. Saldırganların keşif aşamasında toplanan kullanıcı kontrollü girdi noktalarını hedef liste olarak kullanabilecekleri ve kullanıcı kontrollü girdi noktalarının gerçekten de bir açıklık sunup sunmadığını ve sömürülübilir olup olmadığını tespit etmek için kullanıcı kontrollü girdi noktalarına çeşitli yaygın script payload'ları enjekte edebilecekleri ifade edilmektedir.

Saldırının uygulanabilir olup olmadığını deneme aşamasında saldırganların teknik olarak kullanabilecekleri yöntemlerden ilkinde bilinen (elde edilen) kullanıcı kontrollü girdi noktalarını xss'e karşı yoklamak için - eğer mümkünse - ayırt etmek adına unique değerde string içeren XSS probe (XSS yoklayıcı) girdi listesi otomatize bir şekilde kullanabilecekleri söylenmektedir (örn; Burspuite->Intruder gibi).

> Not: Bu xss proble listesi sözlük dosyaları halinde bulunabilir. Bilinen (elde edilen) kullanıcı kontrollü girdi noktalarına otomatize araçlarla fuzzing olarak verilebilir (Burpsuite->Intruder v.b.). Örneğin bu xss probe girdi listesi sözlük dosyasının içerisine <script> bilgem("sgeTest") </script> şeklinde unique değerde string içeren bir xss probe girdi ilavesi yapılabilir. Eğer bu unique değerli xss probe girdisi olduğu gibi encode'lanmadan / filtrelenmeden (yani <script> … </script> halinde) yanıt paketinde geri yansıyorsa XSS zafiyeti var demektir. Burada javascript için anlam ifade etmeyen string değerlerin olması sadece girdinin test amaçlı kullanılması dolayısıyladır. Ancak bu test zafiyetin var olduğunu belirtir. Bu zafiyeti sömürmek için web uygulamanın filtrelerine göre her xss payload'u bu noktada kullanılamayabilir ve xss'in imkan verdiği her türden zarar verilemeyebilir. Fakat bunun yerine xss'in imkan verdiği sınırlı sayıda zararlar verilebilir.

Saldırının uygulanabilir olup olmadığını deneme aşamasında saldırganların teknik olarak kullanabilecekleri alternatif yöntemlerden ikincisinde bilinen (elde edilen) kullanıcı kontrollü girdi noktalarını xss'e karşı yoklamak için girdi noktalarına manuel bir şekilde XSS probe girdi listesindeki girdileri bir proxy aracı ile girebilecekleri ve sonuçlarını proxy aracı ile kayıt altına alarak inceleyebilecekleri söylenmektedir (Örn; Burpsuite->Repeater gibi)

Saldırının uygulanabilir olup olmadığını deneme aşamasında saldırganların teknik olarak kullanabilecekleri alternatif yöntemlerden üçüncüsünde bilinen (elde edilen) kullanıcı kotrollü girdi noktalarını xss'e karşı yoklamak için - eğer mümkünse - ayırt etmek adına unique değerde string içeren xss probe girdi listesindeki girdileri web tarayıcı kullanarak web uygulama arayüzünde yer alan girdi alanlarına girebilecekleri ve sonuçları gözlemleyebilecekleri söylenmektedir.

> Not: Örneğin bu alternatif üçüncü yöntemde <script> alert("BilgemSge"); </script> xss probe girdisi ilaveten kullanılabilir.

Sonuç olarak test aşamasında saldırganlar 3 farklı teknik yolla kullanıcı kontrollü girdi noktalarını xss açıklığı var mı testine tabi tutabilirler. Birincisinde otomatik tool'lar ile xss probe girdi listesini deneyerek ve sonuçları otomatik araçta teker teker inceleyerek, ikincisinde proxy tool'u ile manuel olarak xss probe girdi listesini girerek ve sonuçları proxy araçta teker teker inceleyerek, üçüncüsünde web tarayıcı ile xss probe girdi listesini ekrandaki girdi alanlarına girerek ve sonuçları web tarayıcı ekranında teker teker inceleyerek şeklindedir.

Uyarı: Saldırının uygulanabilir olup olmadığını deneme aşamasında saldırganların teknik olarak kullanabilecekleri alternatif yöntemlerden dördüncüsü vardır, fakat anlaşılamamıştır.

Exploit (Sömürme) Aşaması:

Exploit seçeneğinde saldırganların hangi tür kazançlar elde edilebileceği ve teknik adım olarak bu kazançların nasıl elde edilebileceği ifade edilmektedir.

Saldırganların elde edebilecekleri kazançlar listesinin birincisinde oturum çerezinin çalınması, kullanıcı adı ve parolanın çalınması, kullanıcı sayfa içeriğinin (dom yapısının) çalınması,… örnekleri verilmiştir. Bu kazançları gerçeklemek adına saldırganların teknik olarak kullanabilecekleri yöntemlerden birincisinde; Experiment (deneme) aşamasında elde edilen vektör yoluyla zararlı bir javascript kodu geliştirebilecekleri ve kurbanın tarayıcısına zararlı javascript kodunu yükleyebilecekleri ve döküman bilgisini (dom yapısını) saldırgan web sunucusuna gönderebilecekleri söylenmiştir. Yine bu kazançları gerçeklemek adına saldırganların teknik olarak kullanabilecekleri yöntemlerden alternatif olan ikincisinde; Experiment (deneme) aşamasında elde edilen vektör yoluyla zararlı bir javascript kodu geliştirebilecekleri ve kurbanın tarayıcısında saldırgan web sunucusundan gelen komutlar çalışacak şekilde ayarlayabilecekleri söylenmiştir. Bunu <script src="….saldirganinwebsunucusu.../saldirgan.js> ile yapabilirler. Ya da ilk yöntemde bahsedildiği gibi doğrudan tüm javascript kodlarını girerek de kurbanının web tarayıcısında komutlar çalıştırabilirler.

Saldırganların elde edebilecekleri kazançlar listesinin ikincisinde forceful browsing (güçlü tarama) verilmiştir. Bu kazanç önceki kazançtaki gibi son kullanıcıdan bir şeyler elde etmek yerine son kullanıcıyı kullanarak ilgili web uygulamada eylemler gerçekleştirme ve bir şeyler elde etme veya son kullanıcıyı kullanarak başka web uygulamalarda - örneğin csrf zafiyetli web uygulamalarda - eylemler gerçekleştirme ve bir şeyler elde etme hakkındadır. Bu kazancı gerçeklemek adına saldırganların teknik olarak kullanabilecekleri yöntemlerden birincisinde; Experiment (deneme) aşamasında elde edilen vektör yoluyla zararlı bir javascript kodu geliştirebilecekleri ve kurbanın tarayıcısına zararlı javascript kodunu yükleyebilecekleri ve ilgili web uygulamada eylemler gerçekleştirebilecekleri söylenmektedir. İkinci alternatif yöntemde ise; Experiment (deneme) aşamasında elde edilen vektör yoluyla zararlı bir javascript kodu geliştirebilecekleri ve kurbanın tarayıcısında saldırganın web sunucusundan gelen komutlar çalışacak şekilde ayarlayabilecekleri ve web tarayıcının başka web sitelere (özellikle csrf zafiyetli web sitelere) http talepler yapmasına sebep olabilecekleri söylenmektedir.

Saldırganların elde edebilecekleri kazançlar listesinin üçüncüsünde content spoofing (içerik kandırmaca) verilmiştir. Bu kazanç saldırganın kurbana ilgili uygulamada bir web sayfada almak istediği bilgiyi manipule ederek göstermesi ve kurbanı yanıltması hakkındadır. Bu kazancı gerçeklemek adına saldırganların teknik olarak kullanabilecekleri yöntem için; Experiment (deneme) aşamasında elde edilen vektör yoluyla zararlı bir javascript kodu geliştirebilecekleri ve kurbanın tarayıcısına zararlı javascript kodunu yükleyebilecekleri ve saldırgan tarafından modifiye edilmiş geçersiz bilgiyi kurbanın mevcut sayfasında sergileyebilecekleri söylenmektedir. Bu örneğin bir verinin yanlış veriyle değiştirilmesi sonucu kurbanın yanlış veriyle

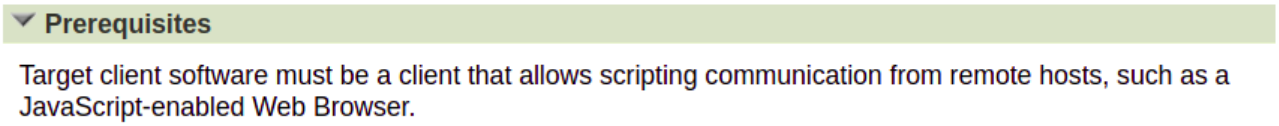
yanıltılması olabilir veya örneğin bir veri eklenmesi sonucu oltalama faaliyeti yürütme olabilir. İkisi de içerik kandırmaca olarak yer alır.

Sonuç olarak sömürme aşamasında 3 çeşit kazanç elde edilebileceği söylenmiştir. Birincisinde kullanıcıdan bir şeyler çalma, ikincisinde kullanıcıyı kullanarak ilgili web uygulamada veya başka web uygulamalarda eylemler gerçekleştirme, üçüncüsünde kurbanın ekranında bilgilerini manipule ederek kurbanı yanıltma şeklindedir.

---

viii) “Prerequisites” Bölümü

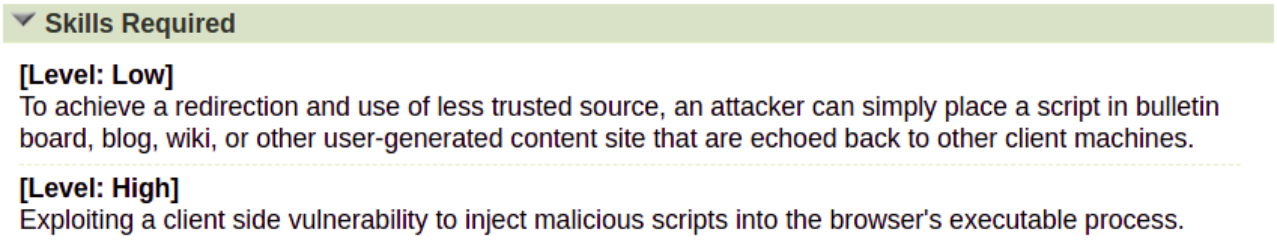
Saldırı deseni sayfalarında daha sonra “Prerequisites (Önkoşullar)” bölümü yer alır. Bu bölümde saldırı deseninin ihtiyaç duyduğu ön koşullar yer alır.



Örneğin CAPEC-63: Cross-Site Scripting saldırı deseni için ön koşul olarak “hedef istemci yazılımı (yani web tarayıcı) uzak web sunucularla script haberleşmesine izin veren (örn; javascript özelliği açık olan) bir istemci (yani web tarayıcı) olmak zorundadır” şartını sunmaktadır. Bu, saldırganların saldırı desenini yürürlüğe koyabilmeleri noktasında gerek duyulan bir koşul konumundadır.

ix) “Skill Required” Bölümü

Saldırı deseni sayfasında daha sonra “Skills Required (Gerekli Yetenekler)” bölümü yer alır. Bu bölümde saldırı deseninin uygulanması noktasında paylaşılan ifadelerdeki kötü faaliyetler için saldırganlarda gerek duyulan bilgi düzeyi bilgisi sunulur.

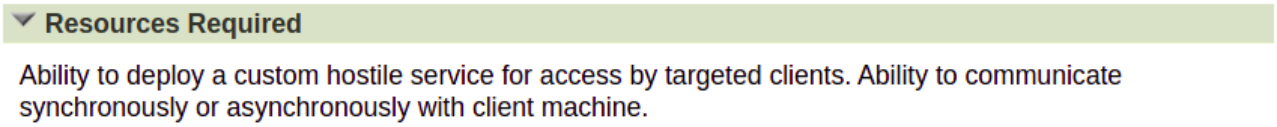


Örneğin Capec-63: Cross-Site Scripting saldırı desenini kullanan saldırganların public içerik platformlarına basitçe bir yönlendime script’i yerleştirmeleri ile diğer istemci makinelere bu script’in yansıması sonucu güvensiz bir web sitesine yönlendirme yapmaları işlemi Low seviyede bir bilgi düzeyi ister denmektedir.

Örneğin Capec-63: Cross-Site Scripting saldırı desenini kullanan saldırganların web tarayıcının çalıştırılabilir process’ine zararlı bir javascript kodu enjekte etmeleri ve bu şekilde istemci taraflı zafiyeti sömürmeleri işlemi yapmaları High seviyede bir bilgi düzeyi ister denmektedir.

x) “Resources Required” Bölümü

Saldırı deseni sayfalarında daha sonra “Resources Required (Gerekli Kaynaklar)” bölümü yer alır. Bu bölümde saldırı desenlerinin uygulanabilmesi noktasında gerek duyulan kaynaklar listelenir.

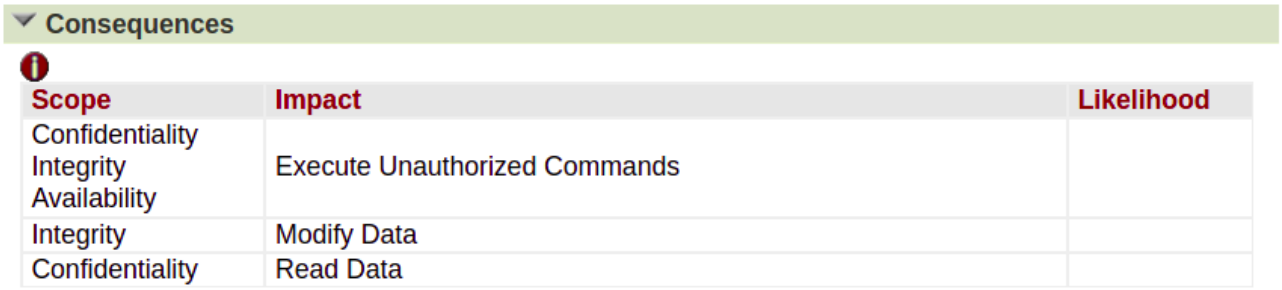


Örneğin CAPEC-63: Cross-Site Scripting saldırı deseninde gerek duyulan kaynaklar için şunlar belirtilmiştir:

- Hedef son kullanıcılarca erişilebilecek, ayağa kaldırılabilir kişisel bir düşman servisi (kişisel bir saldırgan web sunucusu) kaynağı gereklidir.
- Ayrıca saldırgan web sunucusu ile son kullanıcılar arasında synchronously veya asynchronously haberleşme (ajax haberleşmesi) kurulabilir olması kaynağı gereklidir.

xi) “Consequences” Bölümü

Saldırı deseni sayfalarında daha sonra “Consequences (Sonuçlar)” bölümü yer alır. Bu bölümde saldırı deseninin sunduğu etkilerin hangi kapsamda değerlendirilebileceği sonuç olarak raporlanır.



Örneğin CAPEC-63: Cross-Site Scripting saldırı deseninin etkileri Impact sütununda sıralanmıştır ve bu etkilerin hangi kapsamda değerlendirilebilecekleri Scope sütununda gösterilmiştir.

CAPEC-63: Cross-Site Scripting saldırı deseninin yetkisiz kodlar çalıştırma etkisi gizlilik, bütünlük ve erişilebilirlik kapsamlarının üçüne de girmiştir. Çünkü yetkisiz kodlar çalıştırarak gizlilik ihlal edilebilir (örn; çerez çalınabilir, kullanıcı adı ve parola çalınabilir, dom yapısı çalınabilir,...), yetkisiz kodlar çalıştırarak bütünlük ihlal edilebilir (örn; içerikte kandırmaca eklemesi yapılabilir) ve yetkisiz kodlar çalıştırarak erişilebilirlik durdurulabilir (örn; yönlendirme yapılabilir ve esas web sitenin sayfasına erişim sağlanamaz, veya sayfanın kitlenmesine sebep olunabilir, veya sayfa içeriğinin (dom yapısının) silinmesiyle web sitenin sayfasına erişim iptal edilebilir).

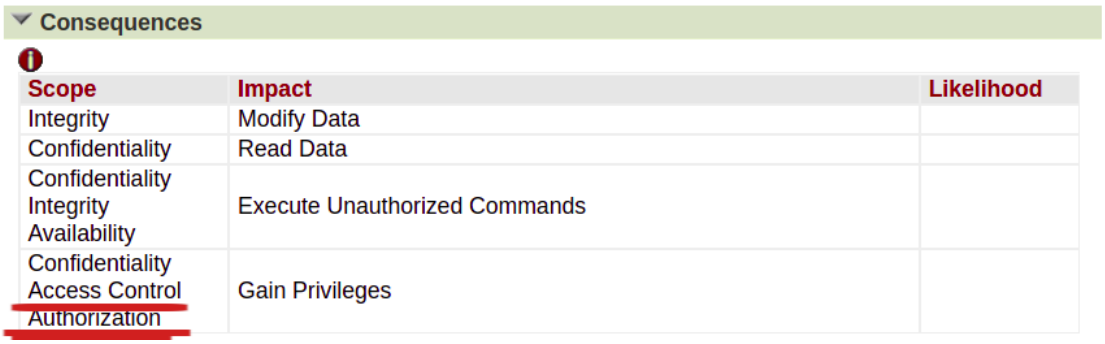
CAPEC-63: Cross-Site Scripting saldırı deseninin “Modify Data (Veriyi Değiştirme)” etkisi Integrity (bütünlük) kriterinin kapsamına girmiştir. Çünkü veri değiştirme yaparak bütünlük bozulabilir (örn; içerikte kandırmaca).

CAPEC-63: Cross-Site Scripting saldırı deseninin “Read Data (Veri Okuma)” etkisi Confidentiality (gizlilik) kriterinin kapsamına girmiştir. Çünkü veri okuma yaparak gizlilik ihlal edilebilir (örn; çerez çalma).

Sonuç olarak bu bölümde ilgili saldırı deseninin etkileri, yani saldırı deseninin yapabildikleri genel bir kategorik isimle isimlendirilir ve bu etkiler hangi kapsamda yer alıyorsa bunlar bu bölümde sonuç olarak raporlanır.

Not:

Kapsam olarak CIA (Confidentiality, Integrity, Availability), yani gizlilik, bütünlük ve erişilebilirlik kullanıldığı gibi farklı kapsamlar da kullanılabilmektedir. Örneğin CAPEC-66 SQL enjeksiyonu saldırı deseninde Authorization ve Access Control kapsamları yer almıştır.



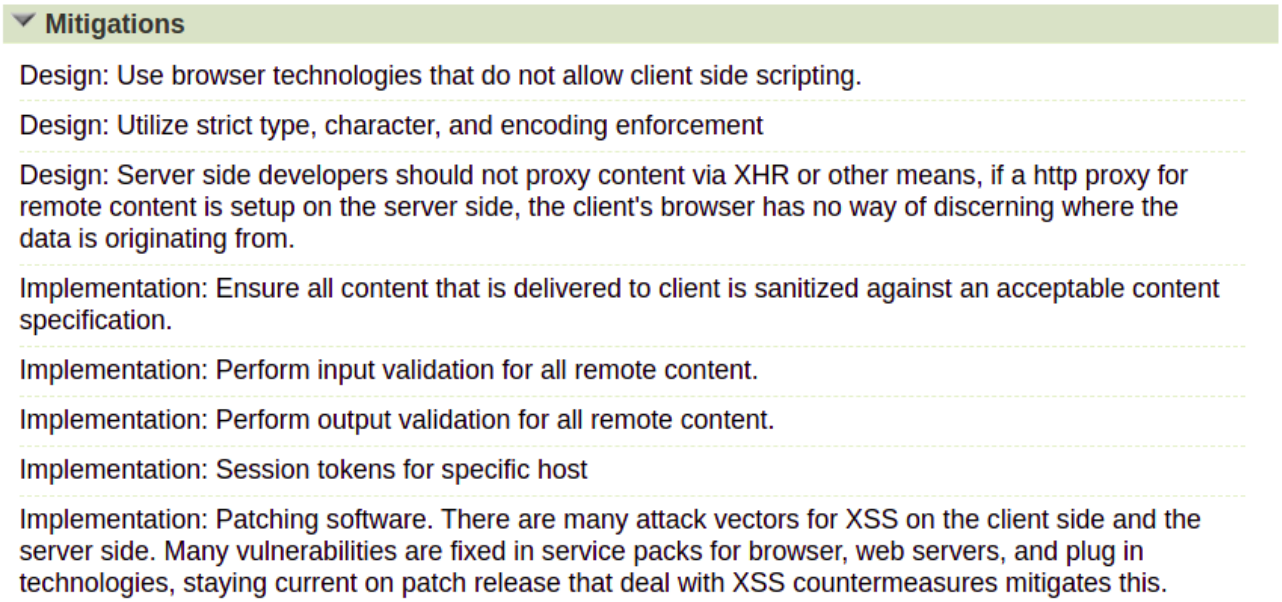
Consequences

Scope	Impact	Likelihood
Integrity	Modify Data	
Confidentiality	Read Data	
Confidentiality Integrity Availability	Execute Unauthorized Commands	
Confidentiality Access Control Authorization	Gain Privileges	

Dolayısıyla kapsam olarak sadece “Gizlilik, Bütünlük, Erişilebilirlik” üçlemesinin kullanımı söz konusu değildir. Farklı kapsamlar da kullanılmaktadır.

xii) “Mitigations” Bölümü

Saldırı deseni sayfalarında daha sonra “Mitigations (Azaltmalar / Hafifletmeler)” bölümü yer alır. Bu bölümde saldırı deseninin uygulanabilirliğini azaltıcı önlemler listelenir.



Mitigations

Design: Use browser technologies that do not allow client side scripting.

Design: Utilize strict type, character, and encoding enforcement

Design: Server side developers should not proxy content via XHR or other means, if a http proxy for remote content is setup on the server side, the client's browser has no way of discerning where the data is originating from.

Implementation: Ensure all content that is delivered to client is sanitized against an acceptable content specification.

Implementation: Perform input validation for all remote content.

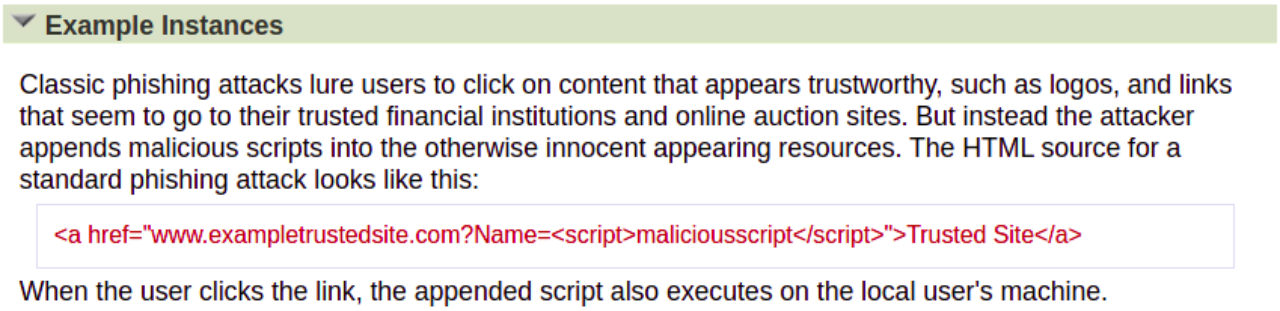
Implementation: Perform output validation for all remote content.

Implementation: Session tokens for specific host

Implementation: Patching software. There are many attack vectors for XSS on the client side and the server side. Many vulnerabilities are fixed in service packs for browser, web servers, and plug in technologies, staying current on patch release that deal with XSS countermeasures mitigates this.

xiii) “Example Instances” Bölümü

Saldırı deseni sayfalarında daha sonra “Example Instances (Örnek Örneklemeler)” bölümü yer alır. Bu bölümde ilgili saldırı deseniyle gerçek dünyada yapılabilecek bir saldırı örneklemesi sunulur.



Örneğin CAPEC-63: Cross-Site Scripting saldırı deseni için gerçek dünyada yapılabilecek bir saldırı örneklemesinde klasik bir oltalama faaliyeti paylaşılmıştır. Bu örneklemede güvenilir bir web adresi linkine saldırgan zararlı bir script kodu eklemiştir ve html kaynak koddan linkin görünen adını güvenilir bir formatta sunmuştur. İlk başta masum görünen bu kaynak saldırganın zararlı kodunu içerdiğinden artık güvenilir bir kaynak olmaktan çıkmıştır.

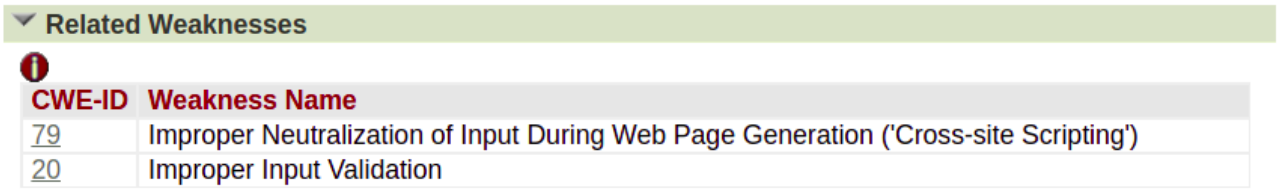
> <a href="www.trustedwebsite.com?Name=<script>maliciousscript</script>">Trusted Site

Kullanıcı bu linke tıkladığında eklenen script kullanıcının yerel makinesinde çalışacaktır.

Sonuç olarak bu bölümde gerçek dünya örnekleri paylaşılır. Bu şekilde saldırı deseni örneklenmiş olur.

xiv) “Related Weaknesses” Bölümü

Saldırı deseni sayfalarında daha sonra “Releated Weaknesses (İlişkili Açıklıklar)” bölümü yer alır. Bu bölümde saldırı deseninin hitap ettiği CWE açıklık türleri listelenir.

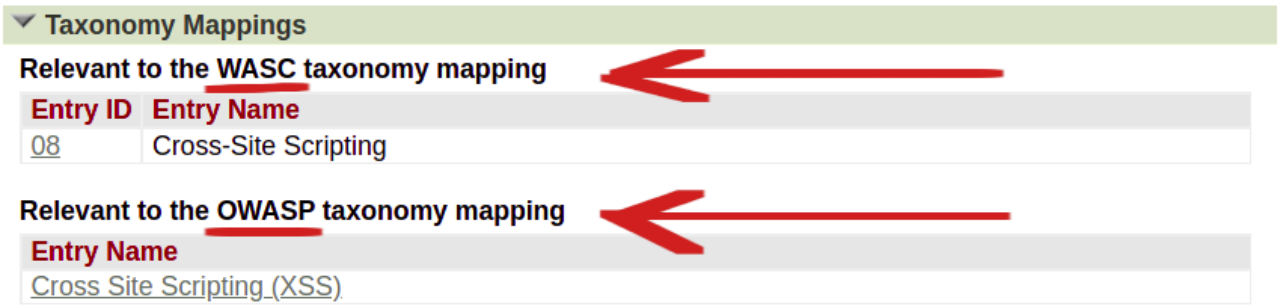


Örneğin CAPEC-63: Cross-Site Scripting saldırı deseni Improper Neutralization of Input During Web Page Generation açıklık türüne hitap etmekteymiş, ilaveten Improper Input Validation açıklık türüne hitap etmekteymiş. Yani birinde input filtrelemesi açıklığına hitap ediyor, birinde ise input doğrulama (beyaz liste, kara liste veya regular expression doğrulaması) açıklık türüne hitap ediyor.

Bu bölümde bir capec saldırı deseninin birden fazla cwe açıklık türüyle eşleştiği görülmektedir. Yani saldırı deseni iki açıklıkla da gerçeklenebilmektedir. Sonuç olarak bu v.b. capec ve cwe ilişkileri bu bölümde paylaşılır.

xv) “Taxonomy Mappings” Bölümü

Saldırı deseni sayfalarında daha sonra “Taxonomy Mappings (Sınıflandırma Eşlemeleri)” bölümü yer alır. Bu bölümde CAPEC’deki saldırı deseninin CAPEC gibi diğer saldırı sınıflandırma standartlarındaki eşlemelerine yer verilir.

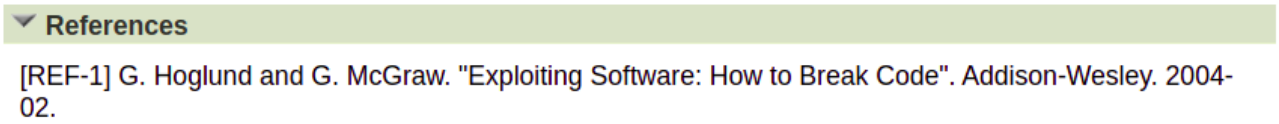


Örneğin CAPEC-63: Cross-Site Scripting saldırı deseni WASC (Web Application Security Consortium) standardında 08 numaralı kayda tekabül etmekteymiş. WASC-8 kaydına gidildiğinde Cross Site Scripting sayfası ve açıklamaları gelecektir. Cross-Site Scripting saldırı deseni OWASP (Open Web Application Project) standardında belirtilen linkteki sayfaya tekabül etmekteymiş.

Sonuç olarak bu bölümde saldırı deseninin diğer sınıflandırma standartlarındaki sayfalarına doğru yönlendirme bilgileri yer alır.

xvi) "References" Bölümü

Saldırı deseni sayfalarında daha sonra "References (Referanslar)" bölümü yer alır. Bu bölümde saldırı deseni sayfasının hazırlanmasında yararlanılan kaynak kitaplara veya makalelelere yer verilir.



Sonuç olarak bu şekilde örneğin kaynak kitapların isimlerini öğrenerek incelemek maksatlı satın alınabilirler. Kaynak kitap keşfi için yararlı olabilir.

xvii) "Content History" Bölümü

Saldırı deseni sayfalarında daha sonra "Content History (İçerik Geçmişi)" bölümü yer alır. Bu bölümde Capec yetkililerinin saldırı deseni sayfasının düzenlenmesine dair geçmiş log'larına yer verilir.

Content History

Submissions		
Submission Date	**Submitter**	**Organization**
2014-06-23	CAPEC Content Team	The MITRE Corporation
Modifications		
Modification Date	**Modifier**	**Organization**
2017-05-01	CAPEC Content Team	The MITRE Corporation
	Updated Activation_Zone, Attack_Prerequisites, Description Summary, Examples-Instances, Payload, Payload_Activation_Impact, Related_Attack_Patterns, Related_Weaknesses, Resources_Required, Typical_Likelihood_of_Exploit	
2020-07-30	CAPEC Content Team	The MITRE Corporation
	Updated Execution_Flow	
2020-12-17	CAPEC Content Team	The MITRE Corporation
	Updated Related_Attack_Patterns, Taxonomy_Mappings	
Previous Entry Names		
Change Date	**Previous Entry Name**	
2017-05-01	Simple Script Injection	

Örneğin CAPEC-63: Cross-Site Scriptin saldırı deseni sayfasında Content History bölümünün en alt satında görülebileceği gibi saldırı deseni başlığı olarak önceden “Simple Script Injection” kullanılmaktaymış bilgisi yer almaktadır.

Yararlanılan Kaynaklar

- https://capec.mitre.org/about/index.html
- https://capec.mitre.org/about/new_to_capec.html
- https://csrc.nist.gov/glossary/term/likelihood
- https://blog.blackswansecurity.com/2020/02/what-is-likelihood/
- https://capec.mitre.org/data/definitions/66.html
- https://capec.mitre.org/data/definitions/3000.html
- https://capec.mitre.org/data/definitions/152.html
- https://capec.mitre.org/about/attack_comparison.html