



## Počítačová obrana a útok

### Protokol z předmětu

**Tématická oblast:** Denial of service (DoS)

**Přednášející:** Ing. Jan Plucar, Ph.D.

**Jméno a číslo studenta:** Adam Šárek (SAR0083)

**Datum vypracování:** 6. 3. 2022

**Zadání:**

Seznamte se s problematikou útoku Denial of Service. Naprogramujte jednoduchou aplikaci, která bude schopna tento útok provést oproti testovacímu Apache serveru.

V aplikaci Wireshark si prohlédněte legitimní provoz, spusťte DoS útok a znovu si prohlédněte provoz přes Wireshark. Porovnejte oba provozy a zodpovězte otázky v závěru.

**Postup řešení:**

- 1) **Implementujte jednoduchou aplikaci, která zamezí legitimnímu provozu testovací DVWA aplikace (Vaše aplikace bude útočit na Apache server). Zdrojový kód vaší aplikace přiložte při odevzdání.**
- 2) **V aplikaci Wireshark si prohlédněte legitimní provoz mezi webovým prohlížečem a DVWA aplikací (IP a TCP hlavičky) – provoz odchytávejte na straně serveru, ne útočníka.**
- 3) **Proved'te DoS útok a otestujte nedostupnost služeb – udělejte screenshot.**
- 4) **V průběhu DoS útoku opět odchyťte provoz přes Wireshark a udělejte screenshot. Opět odchytávejte na straně serveru.**
- 5) **Zpracujte závěr a odpovězte na otázky.**

**Závěr:**

- 1) **Při odevzdání přiložte Vaši aplikaci.**
- 2) **Vytvořte alespoň 2 screenshoty - nedostupná aplikace a DoS útok ve Wiresharku.**
- 3) **Popište, jak probíhá TCP 3-way handshake.**
- 4) **Popište srovnání legitimního provozu a DoS útoku, popište Vaše další zjištění.**
- 5) **Diskutujte možné útoky obrany.**



# Vypracování


## Běžný provoz

Před tím, než zprovozníme DoS útok je dobré se podívat, jak vypadá běžný provoz DVWA. Nejprve se tedy podíváme na výpis programu Wireshark, který zachytává jednotlivé pakety.

ip.src == 192.168.0.157 and tcp.port == 443 and !ssl							
No.	Time	Source	Destination	Protocol	Length	Info	
23	4.936776	192.168.0.157	20.73.130.64	TCP	66	50717 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1	
25	4.960598	192.168.0.157	20.73.130.64	TCP	54	50717 → 443 [ACK] Seq=1 Ack=1 Win=262400 Len=0	
33	4.985611	192.168.0.157	20.73.130.64	TCP	54	50717 → 443 [ACK] Seq=195 Ack=7236 Win=262400 Len=0	
41	5.042487	192.168.0.157	20.73.130.64	TCP	54	50717 → 443 [ACK] Seq=2471 Ack=8631 Win=261120 Len=0	
43	5.043358	192.168.0.157	20.73.130.64	TCP	54	50717 → 443 [FIN, ACK] Seq=2502 Ack=8631 Win=261120 Len=0	
104	19.652138	192.168.0.157	20.73.130.64	TCP	66	50719 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1	
106	19.677281	192.168.0.157	20.73.130.64	TCP	54	50719 → 443 [ACK] Seq=1 Ack=1 Win=262400 Len=0	
114	19.702300	192.168.0.157	20.73.130.64	TCP	54	50719 → 443 [ACK] Seq=195 Ack=7236 Win=262400 Len=0	
121	19.759551	192.168.0.157	20.73.130.64	TCP	54	50719 → 443 [ACK] Seq=2471 Ack=8631 Win=261120 Len=0	
123	19.759987	192.168.0.157	20.73.130.64	TCP	54	50719 → 443 [FIN, ACK] Seq=2502 Ack=8631 Win=261120 Len=0	

Obr. 1 - Ukázka běžného provozu v programu Wireshark

Tato komunikace byla zachycena po zobrazení stránky, kterou je možné vidět na obrázku níže.



[Home](#)[Instructions](#)[Setup / Reset DB](#)  
[Brute Force](#)[Command Injection](#)[CSRF](#)[File Inclusion](#)[File Upload](#)[Insecure CAPTCHA](#)[SQL Injection](#)[SQL Injection \(Blind\)](#)[Weak Session IDs](#)[XSS \(DOM\)](#)[XSS \(Reflected\)](#)[XSS \(Stored\)](#)[CSP Bypass](#)[JavaScript](#)

## Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to practice some of the most common web vulnerabilities, with various levels of difficulty, with a simple straightforward interface.

### General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possible could by using that particular vulnerability.

Please note, there are both documented and undocumented vulnerability with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

DVWA also includes a Web Application Firewall (WAF), PHPIDS, which can be enabled at any stage to further increase the difficulty. This will demonstrate how adding another layer of security may block certain malicious actions. Note, there are also various public methods at bypassing these protections (so this can be seen as an extension for more advanced users)!

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

### WARNING!

Obr. 2 - Ukázka běžného provozu stránky zobrazené v prohlížeči



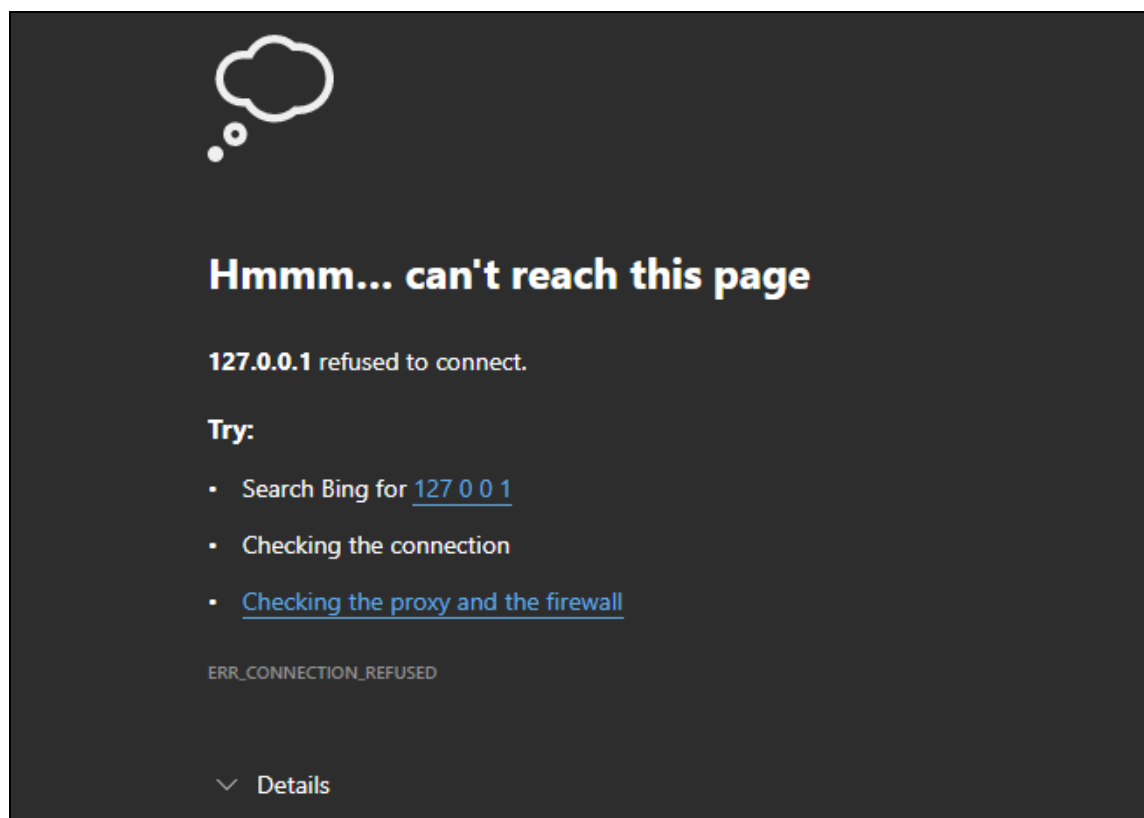
## Provoz během útoku DoS

Nyní využijeme vytvořenou aplikaci s cílem zaútočit na danou IP adresu, tak že ji zahltlíme požadavky. Tento útok by se měl projevit také v programu Wireshark, tak že v danou chvíli útoku začne zachytávat spoustu požadavků, které budou přicházet velmi rychle po sobě, mnohem rychleji než při běžném provozu. Jak to vypadá přímo ve Wiresharku je vidět na obrázku níže.

Current filter: ip.src == 192.168.0.157 and tcp.port == 443 and !ssl							
No.	Time	Source	Destination	Protocol	Length	Info	
30	5.132551	192.168.0.157	192.168.0.139	TCP	66	443 → 60140	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
33	5.133394	192.168.0.157	192.168.0.139	TCP	66	443 → 60141	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
36	5.135053	192.168.0.157	192.168.0.139	TCP	66	443 → 60142	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
39	5.136580	192.168.0.157	192.168.0.139	TCP	66	443 → 60143	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
42	5.138419	192.168.0.157	192.168.0.139	TCP	66	443 → 60144	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
45	5.139787	192.168.0.157	192.168.0.139	TCP	66	443 → 60145	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
48	5.140687	192.168.0.157	192.168.0.139	TCP	66	443 → 60146	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
51	5.141902	192.168.0.157	192.168.0.139	TCP	66	443 → 60147	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
54	5.142775	192.168.0.157	192.168.0.139	TCP	66	443 → 60148	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
57	5.144995	192.168.0.157	192.168.0.139	TCP	66	443 → 60149	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
60	5.146610	192.168.0.157	192.168.0.139	TCP	66	443 → 60150	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
63	5.148349	192.168.0.157	192.168.0.139	TCP	66	443 → 60151	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
66	5.150130	192.168.0.157	192.168.0.139	TCP	66	443 → 60152	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
69	5.151998	192.168.0.157	192.168.0.139	TCP	66	443 → 60153	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
72	5.153970	192.168.0.157	192.168.0.139	TCP	66	443 → 60154	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
75	5.155574	192.168.0.157	192.168.0.139	TCP	66	443 → 60155	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
78	5.157930	192.168.0.157	192.168.0.139	TCP	66	443 → 60156	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
81	5.168220	192.168.0.157	192.168.0.139	TCP	66	443 → 60157	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
84	5.172387	192.168.0.157	192.168.0.139	TCP	66	443 → 60158	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
87	5.173716	192.168.0.157	192.168.0.139	TCP	66	443 → 60159	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
90	5.176220	192.168.0.157	192.168.0.139	TCP	66	443 → 60160	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1

Obr. 3 - Ukázka provozu během útoku DoS v programu Wireshark

Útok se samozřejmě neprojeví jenom ve Wiresharku. Jelikož je daný server zahlcen požadavky, tak není schopen přijímat požadavky uživatele, a proto se není možné na danou stránku již připojit z prohlížeče. V prohlížeči se nám pak zobrazí chybová hláška. Každý prohlížeč vypisuje trochu jinou, nicméně velmi podobnou hlášku. Níže je možné vidět hlášku, kterou vypisuje prohlížeč Microsoft Edge.



Obr. 4 - Ukázka provozu během útoku DoS v prohlížeči Microsoft Edge



## Průběh TCP 3-way handshake

*Handshake*, neboli v překladu „podání ruky“, je proces, který probíhá mezi 2 subjekty pro nastavení parametrů komunikačního spojení. *3-way* poté značí, že daný proces obsahuje celkem 3 cesty mezi klientem a serverem pro zajištění daného nastavení.

Jednotlivé kroky nastavení spojení:

1. Klient, který chce navázat spojení se serverem, posílá TCP segment s nastaveným příznakem SYN (synchronizační sekvenční číslo), kterým informuje server o tom, že by se chtěl připojit a jakým sekvenčním číslem jeho segment začíná. Klient tímto zjišťuje, zda daný server odpovídá a je možné jej použít pro očekávanou komunikaci.
2. Server odpovídá na požadavek klienta pomocí TCP segmentu s nastavenými příznaky SYN+ACK. ACK (potvrzení) označuje odpověď segmentu, který server obdržel a SYN označuje, jakým sekvenčním číslem jeho segment začíná. Cílem tohoto kroku je nejen ujistit klienta, že daný server je schopen odpovídat, ale také zjistit, zda je schopen odpovídat i klient.
3. Klient potvrzuje odpověď serveru odesláním TCP segmentu s nastaveným příznakem ACK, čímž dokazuje, že je také schopen odpovídat a poté již následuje očekávaná komunikace mezi oběma subjekty.

## Srovnání legitimního provozu a DoS útoku

Jak již vychází z ukázky běžného provozu a provozu během útoku DoS, rozdíl mezi oběma případy je v tom, že v rámci legitimního provozu je stránka uživatelům dostupná. Legitimní provoz obvykle obsahuje pouze požadavky uživatelů, kteří tuto stránku chtějí opravdu používat a také tyto požadavky jsou na server posílány v rozumné míře, tedy v intervalech, které produkuje člověk, nikoliv počítač.

Během DoS útoku pak naopak přichází požadavky ve velmi malých intervalech a ve velkém množství, tak že je jimi daný server zahlcen. Server se na všechny požadavky snaží odpovídat, což se projevuje postupným zpomalováním načítání dané stránky, což vede až k tomu, že daná stránka není schopna v rozumném čase odpovědět a stává se pro uživatele nedostupnou.

## Útoky a obrany

### SYN flood

Tento útok spočívá v tom, že útočník velice rychle inicializuje spojení se serverem, avšak toto spojení nedokončuje odesláním TCP segmentu s příznakem ACK, který server očekává. Server tedy marní prostředky čekáním na nedokončené spojení, dokud mu prostředky nedojdou úplně, což vede k tomu, že je server nedostupný pro legitimní provoz.

Obranou proti tomuto útoku je omezení počtu a čekací doby pro spojení z určitého zdroje.

### ICMP flood (Ping flood)

Cílem tohoto útoku je využít daný server jako odesílatele požadavků o ping na různé jiné servery. Tyto servery poté danému odesílateli (cílený server) odesílají odpovědi, čímž tento server mohou zahltit, což samozřejmě není jejich záměr, jelikož pouze odpovídají na to, co byly dotázány.

Obranou proti útoku typu ICMP flood je zakázání funkcionality ICMP na cíleném routeru, počítači a dalších zařízeních. Dalším způsobem může být omezení počtu či velikosti příchozích ICMP zpráv či konfigurace firewallu a přidání filtrování a monitorování paketů.



## Závěr

V rámci tohoto cvičení jsme se seznámili s DoS útoky, vyzkoušeli jsme si implementaci a následný útok na stránku DVWA z předešlého cvičení a odchytili jsme přijímanou komunikaci pomocí programu Wireshark.