Prof. Dr.-Ing. Christof Paar
M. Sc. Felix Hahn
M. Sc. Paul Staat
Dr.-Ing. Steffen Becker

Übung 3

Abgabe: 06.11.2023 **bis** 12:00 Uhr

1. Warm-up: One-Time-Pad 10 Punkte

Das One-Time-Pad (OTP) wurde in der Vorlesung zur Verschlüsselung des binären Alphabets $\Sigma = \{0, 1\}$ eingeführt. Hiermit können Daten von beliebiger Länge bitweise verschlüsselt werden.
Entschlüsseln Sie zum Einstieg den gegebenen Ciphertext c, der einen UTF-8[1]-String aus sieben Zeichen enthält, mit dem ebenfalls angegebenen Schlüssel k. Jedes UTF-8 Zeichen ist nachfolgend durch zwei Hexadezimalzahlen dargestellt.

$$c = \text{0x26c5be19df5cdd}$$

$$k = \text{0x63abca6bb02ca4}$$

Entschlüsseln Sie ein Byte exemplarisch Bit für Bit, danach können Sie die Entschlüsselung direkt (z. B. mit einem Taschenrechner) auf hexadezimalen Werten durchführen. Geben Sie den Klartext hexadezimal und in dekodierter Form (als Wort) an.
Hinweis: Bei UTF-8 handelt es sich um eine Zeichenkodierung, also eine Zuordnung von Zeichen zu ihrer Repräsentation auf Bitebene. In unserem Fall wird jedes Zeichen durch 8 Bit kodiert. In Ihrer Abgabe müssen Sie entsprechend jedes Byte des entschlüsselten Klartextes wieder in den dazu passenden Buchstaben umwandeln.

2. One-Time-Pad 35 Punkte

Sie haben im Moodle eine verschlüsselte Datei gefunden und glauben, dass diese möglicherweise die Lösung für die aktuellen Hausaufgaben enthalten könnte. Laut Beschreibung der Datei handelt es sich um eine PNG-Datei. Es wurde allerdings ein fataler Fehler bei der Verschlüsselung begangen: Um die Benutzung des OTP „einfacher“ zu machen, wurde ein Schlüssel mit einer Länge von 8 Byte eingesetzt, der wiederholt auf den gesamten Klartext angewandt wurde. Das heißt, der Schlüssel wiederholt sich alle 8 Byte.

a) Da Sie bereits wissen, dass es sich um eine PNG-Datei handelt, recherchieren Sie natürlich zunächst einmal, wie genau das PNG-Dateiformat aufgebaut ist. Was stellen Sie dabei in Bezug auf den Anfang der Datei, also den Datei-Header, fest? (5 Pkt.)
Hinweis: Die gesuchten Informationen finden Sie per Internetrecherche, oder aber auch, indem du Sie mehrere unverschlüsselte PNG-Dateien in einem Hex-Editor ihrer Wahl anzeigen lassen.

[1] `https://de.wikipedia.org/wiki/UTF-8`

b) Nun stellt sich die Frage, wie diese Informationen sinnvoll für einen Angriff auf die Verschlüsselung ausgenutzt werden kann. Sie wissen ja bereits, dass der verwendete Schlüssel genau 8 Byte lang ist. Beschreiben Sie in maximal zwei Sätzen, wie ein solcher Angriff aussehen könnte.

(5 Pkt.)

c) Mit dieser Angriffsstrategie ist es nun möglich, den geheimen Schlüssel aus dem verschlüsselten Bild zu extrahieren. Geben Sie den Schlüssel in hexadezimaler Form an und zeigen Sie alle Berechnungen. (10 Pkt.)

d) Nun ist es möglich, die abgefangene Kommunikation komplett zu entschlüsseln. Nutzen Sie den extrahierten Schlüssel um das Bild zu entschlüsseln. Fügen Sie das entschlüsselte Bild der Abgabe hinzu. (15 Pkt.)
Hinweis: Zum Entschlüsseln des Bildes empfehlen wir die Verwendung von CyberChef[2]. Eine Anleitung zu CyberChef finden Sie in Anhang A. Alternativ können Sie auch entweder das bereitgestellte Python[3]-Skript verwenden, oder auf CrypTool[4] zurückgreifen.

3. Vollständige Schlüsselsuche beim One-Time-Pad 15 Punkte

Auf den ersten Blick scheint es möglich, das OTP durch vollständige Schlüsselsuche zu brechen, was einen Widerspruch zu der informationstheoretischen Sicherheit des OTP ist. Gegeben sei eine kurze Nachricht bestehend aus 5 UTF-8 Zeichen, d. h. 40 Bit. Dieser Klartext wurde mit 40 Bit eines OTP verschlüsselt. Beschreiben Sie in max. **3 Sätzen**, warum eine vollständige Schlüsselsuche nicht zum Erfolg führt, obwohl ausreichend Rechenleistung für die Suche zur Verfügung steht.
Bemerkung: Das Paradox muss aufgelöst werden, d. h. Antworten der Form „*Das OTP ist beweisbar sicher, daher funktioniert die vollständige Schlüsselsuche nicht*“ reichen nicht aus.

4. Erweiterter linearer Kongruenzgenerator (LCG) 35 Punkte

Sie haben eine verschlüsselte Nachricht eines Kommilitonen abgefangen. Diese soll einen Link zu geheimen Ressourcen zur Klausurvorbereitung enthalten, also z. B. bisher unveröffentlichte Altklausuren. Diesen Link wollen Sie natürlich sofort entschlüsseln.
Sie wissen, dass ein erweiterter linearer Kongruenzgenerator (LCG) zur Erzeugung eines Schlüsselstroms genutzt wurde. Außerdem wissen Sie, dass der Schlüsselstrom s_i dazu verwendet wurde, den Link in UTF-8 Kodierung zeichenweise zu verschlüsseln, d. h. das Chiffrat y_i ergibt sich für jedes UTF-8 Klartext-Zeichen x_i als bitweises XOR mit s_i. Die erweiterte Variante des LCG berechnet jedes neue Schlüsselstromsymbol s_{i+1} aus den beiden vorherigen Elementen s_i und s_{i-1}. In diesem Fall benötigt der LCG aber auch zwei *seed* Werte s_0 und s_1 sowie insgesamt drei Schlüsselparameter A, B und C und den Modulo m. Die entsprechende Gleichung ist gegeben als:

$$s_{i+1} = A \cdot s_i + B \cdot s_{i-1} + C \bmod m \quad \text{mit } i = 1, 2, \ldots$$

[2] https://gchq.github.io/CyberChef/
[3] https://www.python.org/about/
[4] https://www.cryptool.org/de/

Aufgrund des 8-Bit-ASCII kodierten Klartextes verwenden wir für den erweiterten LCG den Modulus $m = 257$. Die abgefangene Nachricht in hexadezimaler Darstellung lautet wie folgt:

```
5F 23 C8 EE D2 07 8F E0 0E DC 30 03 72 DE FC 37 62 60 1B C1 DA FC 43 4B F9
```

a) Berechnen Sie die Parameter A, B und C durch Aufstellen eines linearen Gleichungssystems. Benutzen Sie Tabelle 1 in Anhang B um die multiplikative Inverse modulo $m = 257$ zu bestimmen. Außerdem wissen Sie, dass es sich um einen Link, der mit der Zeichenkette "https://" beginnt, handelt. (15 Pkt)

b) Wie lautet der vollständige Link? Beschreiben Sie kurz, welche hilfreichen Tipps für Ihr Studium Sie gefunden haben. (10 Pkt.)
Hinweis: Die zweite Frage ist nicht ganz ernst gemeint – beschreiben Sie einfach kurz, was sich hinter dem Link verbirgt!

c) Stellen Sie sich vor, dass ein LCG das Schlüsselstromsymbol s_{i+1} auf Basis der letzten n Elemente $s_i, s_{i-1}, s_{i-2}, \ldots, s_{i-n+1}$ berechnet. Wie viele Parameter $A, B, C, \ldots$ sowie initiale *seed*-Werte werden für diesen LCG benötigt? Wie viele Klartext-Chiffretext-Paare werden für einen Angriff benötigt? (10 Pkt.)

A CyberChef

CyberChef[5] ist ein vom britischen Geheimdienst GCHQ veröffentlichtes OpenSource-Projekt zum Ausführen verschiedenster „Cyber“-Operationen. Der gesamte Quellcode ist auf Github einsehbar und das Tool kann im Browser ausgeführt werden. Unterstützt werden aktuell die Browser *Google Chrome 50+* und *Mozilla Firefox 38+*.
Der Startbildschirm von CyberChef ist in Abbildung 1 dargestellt. Bei **1** wird zunächst der Input als Datei geladen, in unserem Fall also das verschlüsselte Bild. Das mit **2** markierten *Recipe* stellt eine Abfolge von Operationen dar, die nacheinander ausgeführt werden. Ist *Auto Bake* unten rechts aktiviert, werden alle Operationen bei jeder Änderung automatisch erneut ausgeführt. In **3** kann aus verschiedenen Operationen ausgewählt werden. Für die in dieser Übung relevante Aufgabe bietet sich die Verwendung von *Encryption / Encoding* → *XOR* an. Dort ist dann lediglich der Schlüssel einzutragen, die restlichen Einstellungen bleiben unverändert. Zum Anzeigen und Speichern des Bildes sollte unterhalb von *XOR* noch die Operation *Multimedia* → *Render Image* platziert werden. Anschließend wird in **4** das entschlüsselte Bild angezeigt und kann gespeichert werden. Hierbei ist zu beachten, dass die Dateiendung auf *.png* geändert werden sollte.

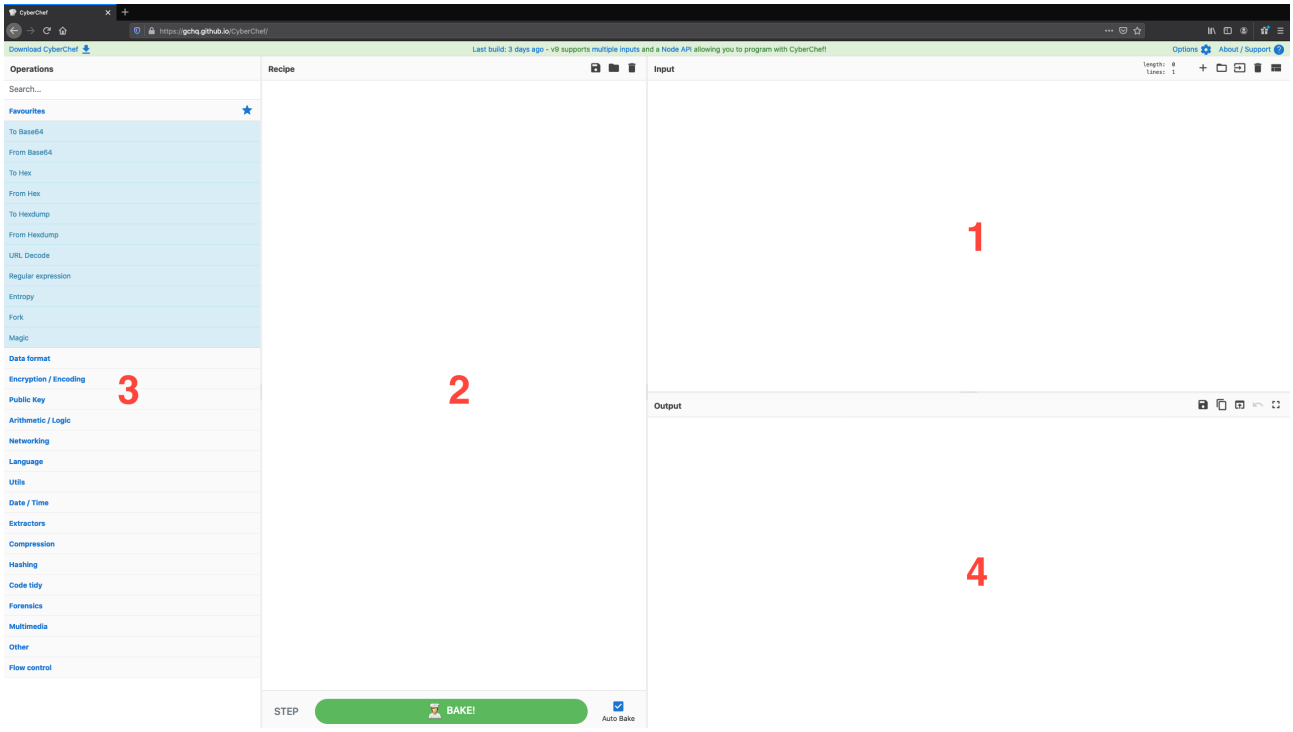

Abbildung 1: Screenshot der Startoberfläche von CyberChef.

[5] https://gchq.github.io/CyberChef/

B Multiplikative Inverse

a	b	$a \cdot b \bmod 257$	a	b	$a \cdot b \bmod 257$	a	b	$a \cdot b \bmod 257$	a	b	$a \cdot b \bmod 257$
1	1	1	2	129	1	3	86	1	4	193	1
5	103	1	6	43	1	7	147	1	8	225	1
9	200	1	10	180	1	11	187	1	12	150	1
13	178	1	14	202	1	15	120	1	16	241	1
17	121	1	18	100	1	19	230	1	20	90	1
21	49	1	22	222	1	23	190	1	24	75	1
25	72	1	26	89	1	27	238	1	28	101	1
29	195	1	30	60	1	31	199	1	32	249	1
33	148	1	34	189	1	35	235	1	36	50	1
37	132	1	38	115	1	39	145	1	40	45	1
41	163	1	42	153	1	43	6	1	44	111	1
45	40	1	46	95	1	47	175	1	48	166	1
49	21	1	50	36	1	51	126	1	52	173	1
53	97	1	54	119	1	55	243	1	56	179	1
57	248	1	58	226	1	59	61	1	60	30	1
61	59	1	62	228	1	63	102	1	64	253	1
65	87	1	66	74	1	67	234	1	68	223	1
69	149	1	70	246	1	71	181	1	72	25	1
73	169	1	74	66	1	75	24	1	76	186	1
77	247	1	78	201	1	79	244	1	80	151	1
81	165	1	82	210	1	83	96	1	84	205	1
85	127	1	86	3	1	87	65	1	88	184	1
89	26	1	90	20	1	91	209	1	92	176	1
93	152	1	94	216	1	95	46	1	96	83	1
97	53	1	98	139	1	99	135	1	100	18	1
101	28	1	102	63	1	103	5	1	104	215	1
105	164	1	106	177	1	107	245	1	108	188	1
109	224	1	110	250	1	111	44	1	112	218	1
113	116	1	114	124	1	115	38	1	116	113	1
117	134	1	118	159	1	119	54	1	120	15	1
121	17	1	122	158	1	123	140	1	124	114	1
125	220	1	126	51	1	127	85	1	128	255	1
129	2	1	130	172	1	131	206	1	132	37	1
133	143	1	134	117	1	135	99	1	136	240	1
137	242	1	138	203	1	139	98	1	140	123	1
141	144	1	142	219	1	143	133	1	144	141	1
145	39	1	146	213	1	147	7	1	148	33	1
149	69	1	150	12	1	151	80	1	152	93	1
153	42	1	154	252	1	155	194	1	156	229	1
157	239	1	158	122	1	159	118	1	160	204	1
161	174	1	162	211	1	163	41	1	164	105	1
165	81	1	166	48	1	167	237	1	168	231	1
169	73	1	170	192	1	171	254	1	172	130	1
173	52	1	174	161	1	175	47	1	176	92	1
177	106	1	178	13	1	179	56	1	180	10	1
181	71	1	182	233	1	183	191	1	184	88	1
185	232	1	186	76	1	187	11	1	188	108	1
189	34	1	190	23	1	191	183	1	192	170	1
193	4	1	194	155	1	195	29	1	196	198	1
197	227	1	198	196	1	199	31	1	200	9	1
201	78	1	202	14	1	203	138	1	204	160	1
205	84	1	206	131	1	207	221	1	208	236	1
209	91	1	210	82	1	211	162	1	212	217	1
213	146	1	214	251	1	215	104	1	216	94	1
217	212	1	218	112	1	219	142	1	220	125	1
221	207	1	222	22	1	223	68	1	224	109	1
225	8	1	226	58	1	227	197	1	228	62	1
229	156	1	230	19	1	231	168	1	232	185	1
233	182	1	234	67	1	235	35	1	236	208	1
237	167	1	238	27	1	239	157	1	240	136	1
241	16	1	242	137	1	243	55	1	244	79	1
245	107	1	246	70	1	247	77	1	248	57	1
249	32	1	250	110	1	251	214	1	252	154	1
253	64	1	254	171	1	255	128	1	256	256	1

Tabelle 1: Multiplikative Inverse modulo $m = 257$.