Rechnersicherheit - SS17 - FU Berlin

Gruppe:

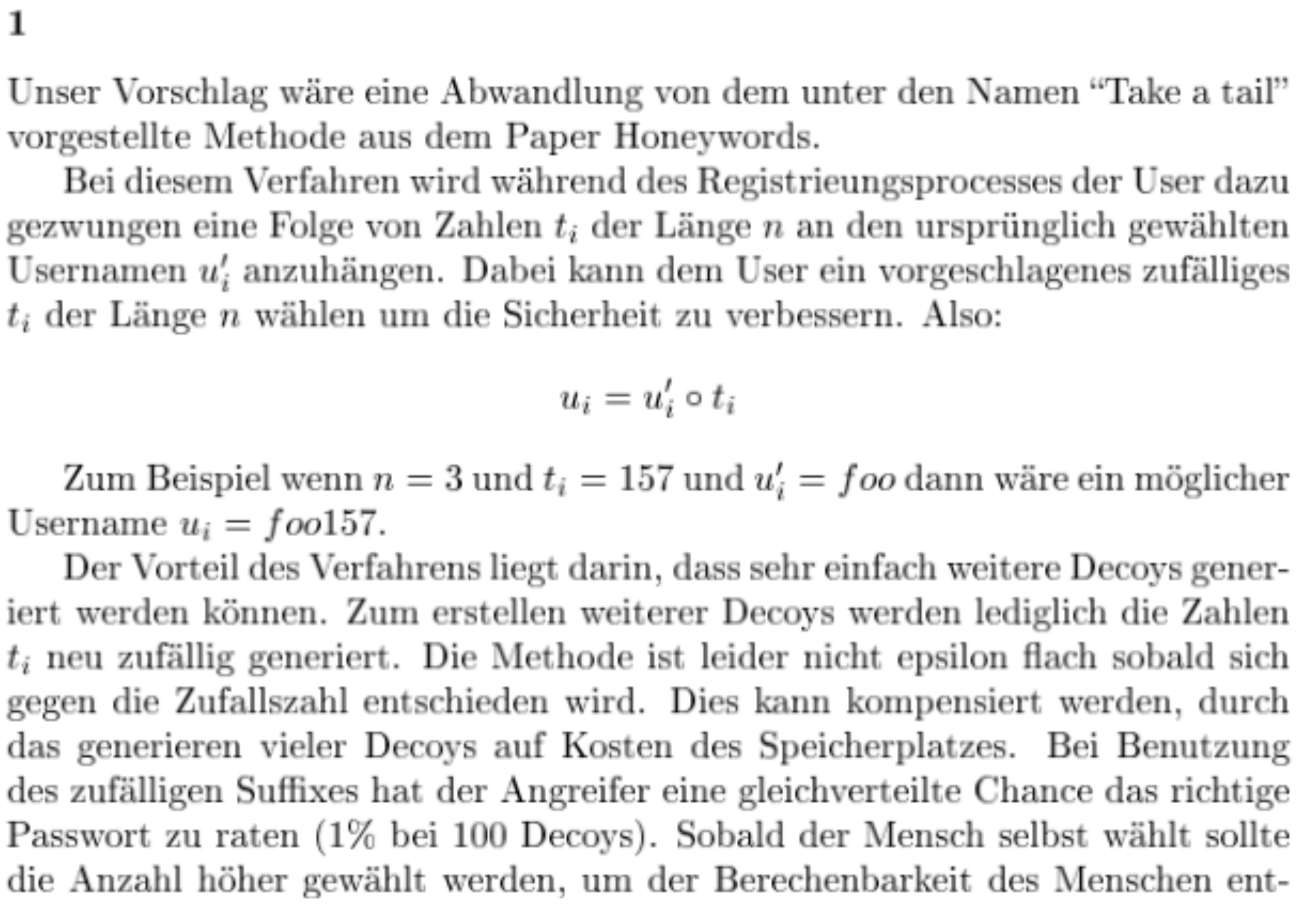
- Jakob Krause, Master Informatik, 4573260
- Ramdane Sennoun, Master Informatik, 4304440

Papers

- OMEN: Faster Password Guessing Using an Ordered Markov Enumerator
- Martin Hellman: A cryptoanalytic time-memory trade-off)
- Philippe Cechslim: Making a faster cryptoanalytic time-memory trade-off
- Narayanan, Shmatrikov: Fast dictionary attacks on passwords using time-memory-trade-off

Assignment 6

Exercise 17 (Honeywords (3+2 Points)).



1

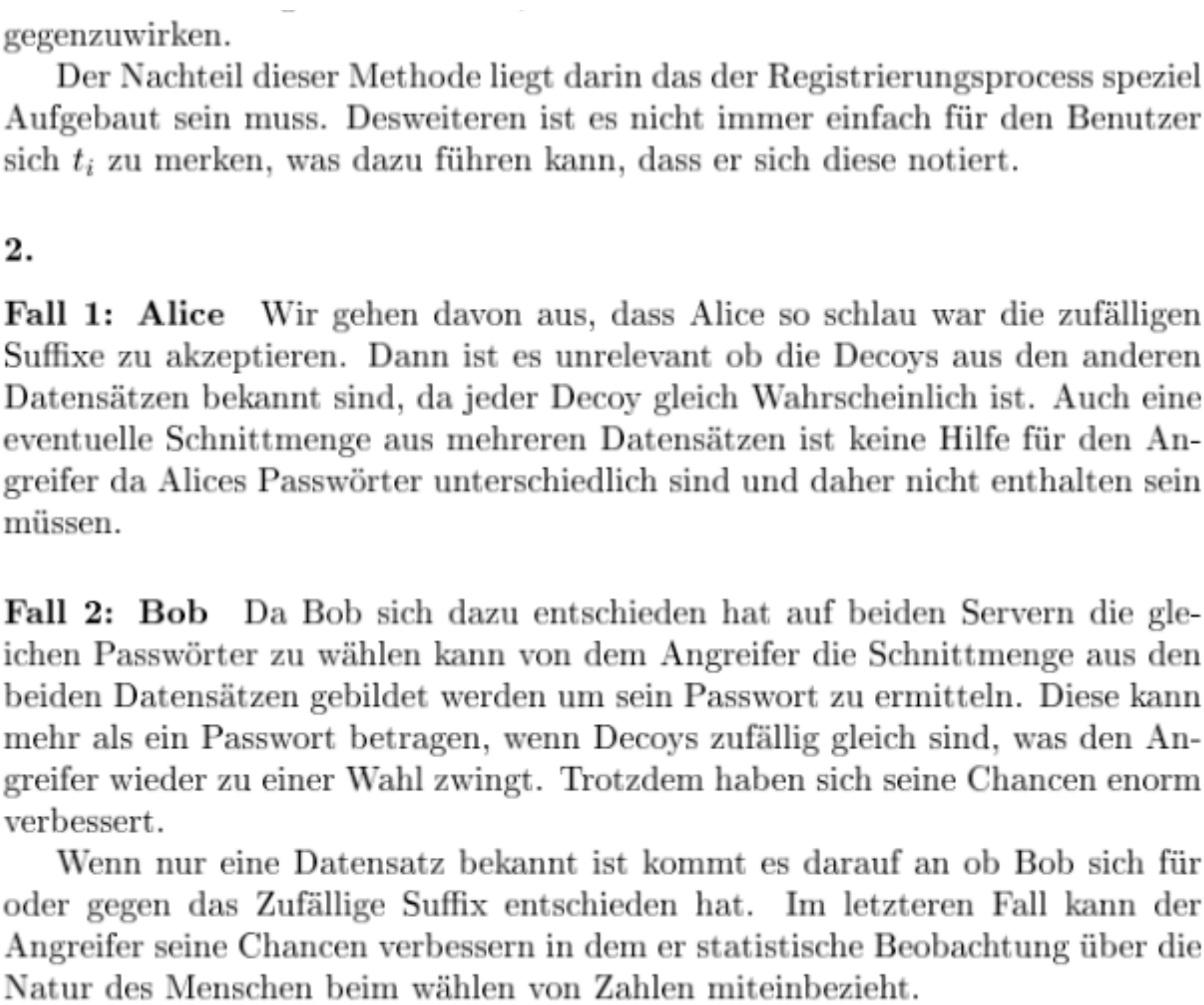
Unser Vorschlag wäre eine Abwandlung von dem unter den Namen “Take a tail” vorgestellte Methode aus dem Paper Honeywords.

Bei diesem Verfahren wird während des Registrieungsprocesses der User dazu gezwungen eine Folge von Zahlen t_i der Länge n an den ursprünglich gewählten Usernamen u'_i anzuhängen. Dabei kann dem User ein vorgeschlagenes zufälliges t_i der Länge n wählen um die Sicherheit zu verbessern. Also:

$$u_i = u'_i \circ t_i$$

Zum Beispiel wenn $n = 3$ und $t_i = 157$ und $u'_i = foo$ dann wäre ein möglicher Username $u_i = foo157$.

Der Vorteil des Verfahrens liegt darin, dass sehr einfach weitere Decoys generiert werden können. Zum erstellen weiterer Decoys werden lediglich die Zahlen t_i neu zufällig generiert. Die Methode ist leider nicht epsilon flach sobald sich gegen die Zufallszahl entschieden wird. Dies kann kompensiert werden, durch das generieren vieler Decoys auf Kosten des Speicherplatzes. Bei Benutzung des zufälligen Suffixes hat der Angreifer eine gleichverteilte Chance das richtige Passwort zu raten (1% bei 100 Decoys). Sobald der Mensch selbst wählt sollte die Anzahl höher gewählt werden, um der Berechenbarkeit des Menschen ent-



gegenzuwirken.

Der Nachteil dieser Methode liegt darin das der Registrierungsprocess speziel Aufgebaut sein muss. Desweiteren ist es nicht immer einfach für den Benutzer sich t_i zu merken, was dazu führen kann, dass er sich diese notiert.

2.

Fall 1: Alice Wir gehen davon aus, dass Alice so schlau war die zufälligen Suffixe zu akzeptieren. Dann ist es unrelevant ob die Decoys aus den anderen Datensätzen bekannt sind, da jeder Decoy gleich Wahrscheinlich ist. Auch eine eventuelle Schnittmenge aus mehreren Datensätzen ist keine Hilfe für den Angreifer da Alices Passwörter unterschiedlich sind und daher nicht enthalten sein müssen.

Fall 2: Bob Da Bob sich dazu entschieden hat auf beiden Servern die gleichen Passwörter zu wählen kann von dem Angreifer die Schnittmenge aus den beiden Datensätzen gebildet werden um sein Passwort zu ermitteln. Diese kann mehr als ein Passwort betragen, wenn Decoys zufällig gleich sind, was den Angreifer wieder zu einer Wahl zwingt. Trotzdem haben sich seine Chancen enorm verbessert.

Wenn nur eine Datensatz bekannt ist kommt es darauf an ob Bob sich für oder gegen das Zufällige Suffix entschieden hat. Im letzteren Fall kann der Angreifer seine Chancen verbessern in dem er statistische Beobachtung über die Natur des Menschen beim wählen von Zahlen miteinbezieht.

Exercise 18 (x86 assembly recap (2 Points)).

See files.

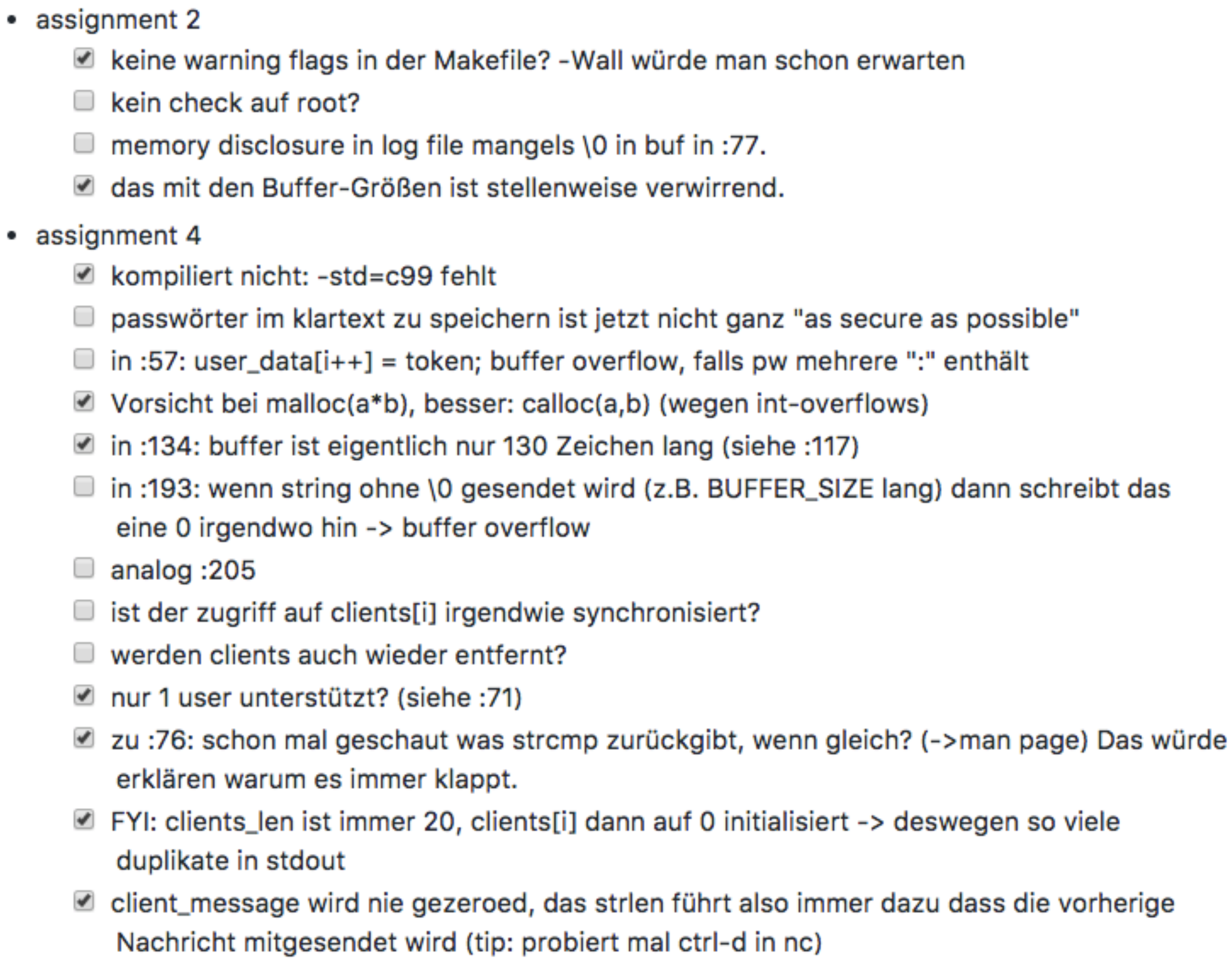
Exercise 19 (Case study µ-shout (II): Secure C Coding (2 Points + 1 Bonus)).

Part 1 Code:

See files for code.

Part 2 Notes:

This is a summary of the fixed issues. Not all issues were fixed yet. We fix more issues as we go.



- assignment 2
  - ☑ keine warning flags in der Makefile? -Wall würde man schon erwarten
  - ☐ kein check auf root?
  - ☐ memory disclosure in log file mangels \0 in buf in :77.
  - ☑ das mit den Buffer-Größen ist stellenweise verwirrend.
- assignment 4
  - ☑ kompiliert nicht: -std=c99 fehlt
  - ☐ passwörter im klartext zu speichern ist jetzt nicht ganz "as secure as possible"
  - ☐ in :57: user_data[i++] = token; buffer overflow, falls pw mehrere ":" enthält
  - ☑ Vorsicht bei malloc(a*b), besser: calloc(a,b) (wegen int-overflows)
  - ☑ in :134: buffer ist eigentlich nur 130 Zeichen lang (siehe :117)
  - ☐ in :193: wenn string ohne \0 gesendet wird (z.B. BUFFER_SIZE lang) dann schreibt das eine 0 irgendwo hin -> buffer overflow
  - ☐ analog :205
  - ☐ ist der zugriff auf clients[i] irgendwie synchronisiert?
  - ☐ werden clients auch wieder entfernt?
  - ☑ nur 1 user unterstützt? (siehe :71)
  - ☑ zu :76: schon mal geschaut was strcmp zurückgibt, wenn gleich? (->man page) Das würde erklären warum es immer klappt.
  - ☑ FYI: clients_len ist immer 20, clients[i] dann auf 0 initialisiert -> deswegen so viele duplikate in stdout
  - ☑ client_message wird nie gezeroed, das strlen führt also immer dazu dass die vorherige Nachricht mitgesendet wird (tip: probiert mal ctrl-d in nc)

Some of the variable names changed like `buf`. Also added more descriptive named constants. Still not an ideal solution. Password still need to be hashed. Variable accesses need to be synchronized.

Another point: we need to implement a dynamic array solution for adding and removing connected clients. The access for that array also will need to be synchronized.

Part 3 Secure C Coding Rules:

We decided to take a look at rule 6: Rule 06. Arrays (ARR) under ARR32-C. Ensure size arguments for variable length arrays are in a valid range.

We ensure that the passed index (i.e. adding `connected_client` the `clients` array) is in valid range.

We also decided to take a look at rule 48: Rule 48. Miscellaneous (MSC) under MSC37-C. Ensure that control never reaches the end of a non-void function.

In our `UShout` code all non void methods are returning a valid value in any case.

Assignment 5

Exercise 13 (Rainbow tables (3 Points)).

$$\begin{aligned} R(91) &= 9126 \\ H(9126) &= 17 \\ R(17) &= 1742 \\ H(1742) &= 59 \end{aligned}$$

Ein Endpunkt wurde gefunden nun faengt man an von Startpoint zu hashen

$$\begin{aligned} H(1234) &= 46 \\ R(46) &= 4671 \\ H(4671) &= 17 \end{aligned}$$

Der Hashwert ist schon bekannt daher macht es keinen Sinn hier weiter zu hashen. Stattdessen machen wir weiter bei $H(1742) = 56$

$$\begin{aligned} R(59) &= 5984 \\ H(5984) &= 33 \\ R(33) &= 3368 \\ H(3368) &= 91 \end{aligned}$$

Die gesuchte Eingabe ist demnach 3368.

Exercise 14 (x86 assembly recap (1+2 Points)).

Das E als Prefix gibt an das es sich um eine Registergröÿe von 32-Bit handelt.

ESP (Stack Pointer)

Das Register zeigt auf die Spitze des Stacks.

EBP (Stack Base Pointer)

Das Register zeigt auf den Boden des Stacks.

EIP (Instruction Pointer)

Das Register speichert die Adresse der folgende CPU Anweisung es sei den Branching ist möglich.

Exercise 15 (Common mistakes / lessons learned (4 Points)).

We implemented the C-program and compiled it with `gcc` on our Debian machine.

Running the program (`./a.out`) prompts the user with `Password:`

Typing `password` as the password returns `Access Denied`.

An example password, that does not start with "password" is "helloworld", actually any string that has at least 10 characters. The programm has the following line of code: `char buffer [9];`. There is not enough space in `buffer`, since one extra char is needed for the terminating null character which causes overflowing bytes get written over.

```
Password OK
Segmentation fault
```

`objdump` displays information from an object or program including functions and read-only data. Option `-d` disassembles the data to make it more accessible.

See the `objdump_output.txt` file.

`gdb` (C debugger) was used to debug the c program: `gdb a.out`.

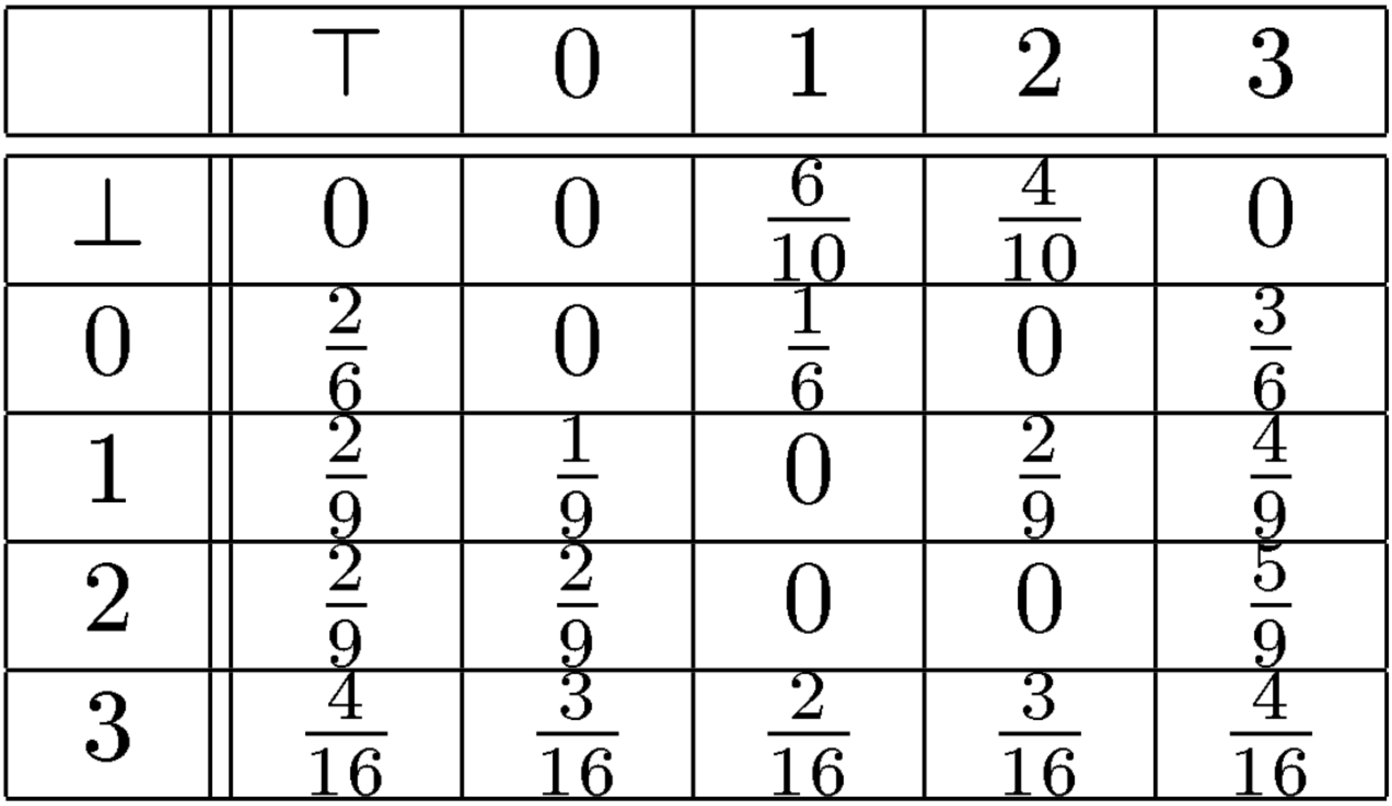
Exercise 16 (Keeping your systems secure (Bonus: 1 Points)).

not solved

Assignment 3

Exercise 5 (Markov-Generator (5 Points)).

first-order Markov generator



	$\top$	0	1	2	3
$\bot$	0	0	$\frac{6}{10}$	$\frac{4}{10}$	0
0	$\frac{2}{6}$	0	$\frac{1}{6}$	0	$\frac{3}{6}$
1	$\frac{2}{9}$	$\frac{1}{9}$	0	$\frac{2}{9}$	$\frac{4}{9}$
2	$\frac{2}{9}$	$\frac{2}{9}$	0	0	$\frac{5}{9}$
3	$\frac{4}{16}$	$\frac{3}{16}$	$\frac{2}{16}$	$\frac{3}{16}$	$\frac{4}{16}$

second-order Markov generator

	$\top$	0	1	2	3
$\perp$	0	0	$\frac{6}{10}$	$\frac{4}{10}$	0
00	0	0	0	0	0
01	1	0	0	0	0
02	0	0	0	0	0
03	$\frac{2}{3}$	0	0	0	$\frac{1}{3}$
10	0	0	0	0	1
11	0	0	0	0	0
12	$\frac{1}{2}$	$\frac{1}{2}$	0	0	0
13	0	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{2}{4}$
20	$\frac{1}{2}$	0	0	0	$\frac{1}{2}$
21	0	0	0	0	0
22	0	0	0	0	0
23	$\frac{1}{5}$	$\frac{1}{5}$	0	$\frac{2}{5}$	$\frac{1}{5}$
30	$\frac{1}{3}$	0	$\frac{1}{3}$	0	$\frac{1}{3}$
31	$\frac{1}{2}$	0	0	$\frac{1}{2}$	0
32	$\frac{1}{3}$	$\frac{1}{3}$	0	0	$\frac{1}{3}$
33	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	0

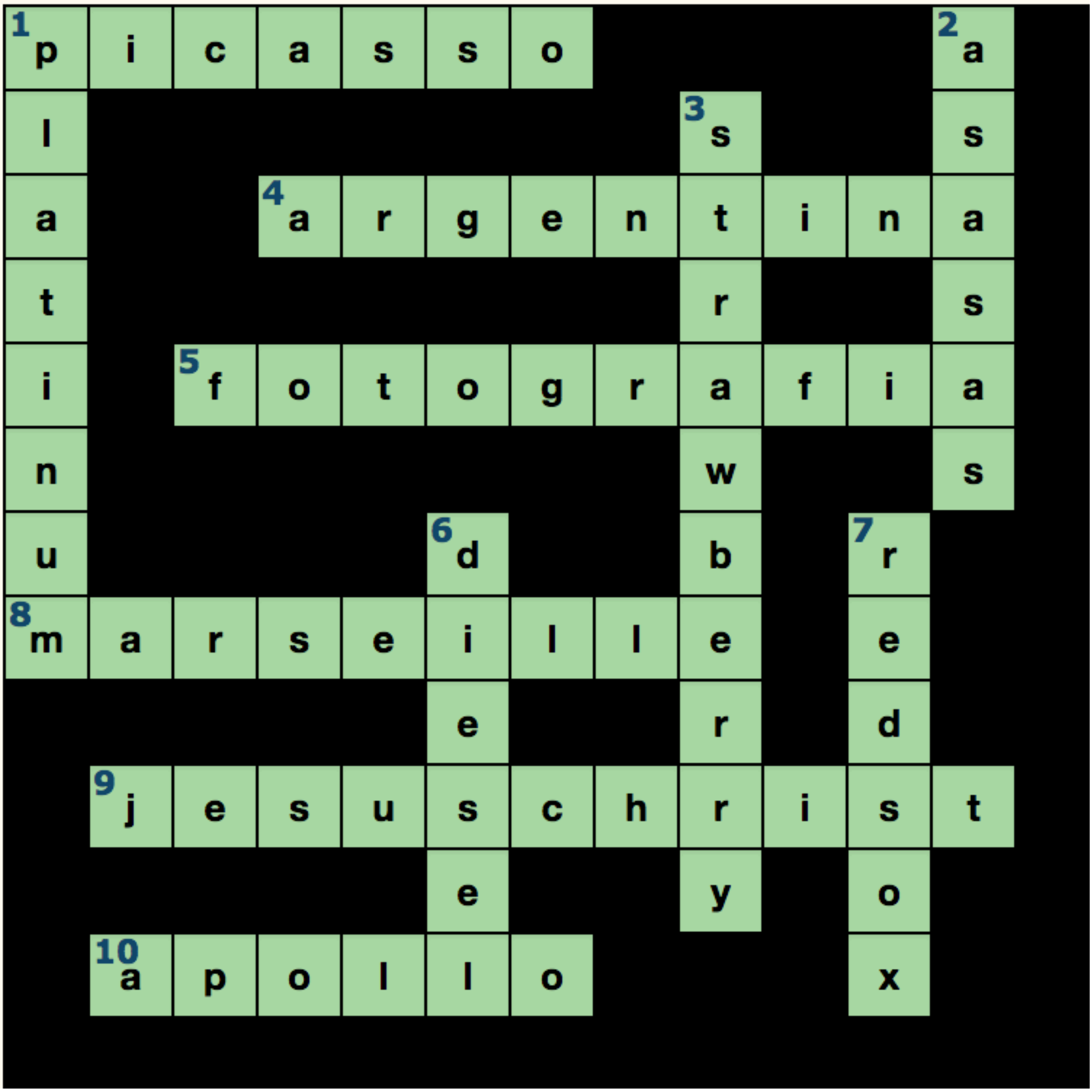
Give one 4-digit PIN number that is generated by your first-order Markov generator but not by your second-order generator and calculate it’s proba- bility.

Die PIN 2333 kann von dem 2. Model nicht generiert werden. Die Wahrschein- lichkeit dieser PIN beträgt:

$$\frac{4}{10} * (\frac{4}{16})^3 = 0.00625$$

Exercise 6 (Bad Password Practice (1+2 Points)).

6.1 Solution (Password popularity: 701-800)



1 p	i	c	a	s	s	o					2 a
l								3 s			s
a			4 a	r	g	e	n	t	i	n	a
t								r			s
i		5 f	o	t	o	g	r	a	f	i	a
n								w			s
u					6 d			b		7 r	
8 m	a	r	s	e	i	l	l	e		e	
					e			r		d	
	9 j	e	s	u	s	c	h	r	i	s	t
					e			y		o	
	10 a	p	o	l	l	o				x	

6.2 What did Adobe do wrong?

Adobe used 3DES, a block cipher, in ECB mode to encrypt the passwords. Therefore they made it easier for bruteforce methods because it is a fast encryption method. Their chose method requires a cipher password that needs to be in the system all the time. Therefor, if that cipher password gets published somehow, all the passwords can be revealed. It also gives information about the password length.

Exercise 7 (Randomized Response (2+1 Points)).

Exercise 8 (Keeping your systems secure (Bonus: 1 Points)).

Debian

Since April 29th, 11.59pm there were 13 debian security announcemenets as of this writing.

- [SECURITY] [DSA 3840-1] mysql-connector-java Moritz Muehlenhoff
- [SECURITY] [DSA 3841-1] libxstream-java Moritz Muehlenhoff
- [SECURITY] [DSA 3842-1] tomcat7 security update Sebastien Delafond
- [SECURITY] [DSA 3843-1] tomcat8 security update Sebastien Delafond
- [SECURITY] [DSA 3844-1] tiff security update Moritz Muehlenhoff
- [SECURITY] [DSA 3845-1] libtirpc security update Moritz Muehlenhoff
- [SECURITY] [DSA 3846-1] libytnef security update Sebastien Delafond
- [SECURITY] [DSA 3847-1] xen security update Moritz Muehlenhoff
- [SECURITY] [DSA 3848-1] git security update Salvatore Bonaccorso
- [SECURITY] [DSA 3849-1] kde4libs security update Salvatore Bonaccorso
- [SECURITY] [DSA 3850-1] rtmpdump security update Moritz Muehlenhoff
- [SECURITY] [DSA 3851-1] postgresql-9.4 security update Moritz Muehlenhoff
- [SECURITY] [DSA 3852-1] squirrelmail security update Sebastien Delafond

The list was extracted from https://lists.debian.org/debian-security-announce/2017/threads.html

[SECURITY] [DSA 3848-1] git security update

The Git git-shell, a restricted login shell for Git-only SSH access, allows a user to run an interactive pager by causing it to spawn `git upload-pack --help`.

They recommend to upgrade the git packages.

OpenBSD

Since April 29th, 11.59pm there were 7 OpenBSD 6.1 security announcemenets as of this writing.

- 001: INTEROPERABILITY FIX: May 2, 2017 All architectures dhcpd unconditionally echoed the client identifier, preventing some devices from acquiring a lease. A source code patch exists which remedies this problem.
- 002: SECURITY FIX: May 2, 2017 amd64 vmm mismanaged floating point contexts. A source code patch exists which remedies this problem.
- 003: SECURITY FIX: May 2, 2017 All architectures A consistency check error could cause programs to incorrectly verify TLS certificates when using callbacks that always return 1. A source code patch exists which remedies this problem.
- 004: RELIABILITY FIX: May 2, 2017 All architectures softraid was unable to create usable concat volumes because it always set the size of the volume to zero sectors. A source code patch exists which remedies this problem.
- 005: RELIABILITY FIX: May 6, 2017 All architectures Expired pf source tracking entries never got removed, leading to memory exhaustion. A source code patch exists which remedies this problem.

- 006: RELIABILITY FIX: May 8, 2017 All architectures Incorrect DTLS cookie handling can result in a NULL pointer dereference. A source code patch exists which remedies this problem.
- 007: SECURITY FIX: May 13, 2017 All architectures Heap-based buffer overflows in freetype can result in out-of-bounds writes. A source code patch exists which remedies this problem.

The list was extracted from https://www.openbsd.org/errata61.html

Assignment 2

Exercise 3 (Case study µ-shout (I): a small echo server (6 Points)).

- see files

Exercise 4 (Real-world de-anonymization 2+2 Points).

Read https://33bits.org/2008/11/12/57/ and download the current “Loan Data” history from https://www.lendingclub.com/info/statistics.action.

1. **The blog post from above is quite old. Which steps can you identify that Lending Club has taken to “anonymize” the data?**

   According to their privacy policy they "protect personal information from unauthorized access and use by using security measures that comply with federal law. These measures include computer safeguards and secured files and buildings."

   Their security notice says: "Physical safeguards like access badges and a video monitoring system protect Lending Club from unauthorized access. Administrative safeguards like a security training program, employment background checks, and access controls ensure that your information is accessed only by trained and trusted staff based on business needs to provide services to you. Technical safeguards like perimeter security, industry standard encryption, two-factor authentication, and continuous monitoring ensure that your information is protected with layered and complementary controls. We review and adjust security safeguards periodically based on ongoing risk assessment. However, despite our substantial investment in protecting your personal information, ultimately no institution can guarantee its absolute security."

   They use methods like Encryption & website certificates, Session time-outs, passwords and Identity theft protection to protect personal information.

   We downloaded the 2017 Q1 loan data and the data dictionary to understand the meaning of each field. So, to "anonymize" their customers data they just didn't include fields like first*name and last*name. That does not matter because they can be "de-anonymized" with data fields like location or job and searching that in combinatino with their screenname and other sensitive information that can be taken from the data. Also, to "hide" their location they xx-ed out the last two digits of the zip code.

Compared to an older data set it seems like they removed the detailed loan description and categorized the loan instead, i.e. "Dept consolidation" or "major purchase".

We also don't see the screen name anymore like the blog post decribed it.

2. **Find the identity (name or social-media account or homepage, etc) of one lender and describe your process (note that some lenders are easier to identify than others).**

   We downloaded an older data set (2007 - 2011). We were looking for unique job descriptions and found this entry:

   ```
   ,"","7000","7000","7000"," 60 months"," 15.96%","170.08","C","C5","Southern St
   ar Photography","8 years","RENT","47004","Not Verified","Dec-2011","Fully Paid
   ","n",""," Borrower added on 12/18/11 > I am planning on using the funds to p
   ay off two retail credit cards with 24.99% interest rates, as well as a major
   bank credit card with a 18.99% rate.  I pay all my bills on time, looking for
   a lower combined payment and lower monthly
   ```

   All we needed to do was googling Southern Star Photography for North Carolina and we are pretty sure we found the lender. Even pictures from his family were accessible:

   [His public Facebook page]()

   For the most recent data set we tried a similar approach. We searched for entries that contained LLC and came across this entry:

   ```
   ,"","21300","21300","21300"," 36 months"," 19.99%","791.48","D","D4","Kennedy
   Transport, LLC","3 years","MORTGAGE","47360","Verified","Feb-2017","Current","
   n","","","debt_consolidation","Debt consolidation","712xx","LA","24.35","3","A
   ug-1985","0","18","","9","0","18327","79%","19","f","20419.41","20419.41","153
   5.65","1535.65","880.59","655.06","0.0","0.0","0.0","Apr-
   ```

   We searched the company name and found these entries which matched the location "LA" and which seems to be operated by one person:

   - [Company]()
   - [Empty Facebook Page]()
   - [Company Profile on movers.com]() which has address and contact information which will lead to the person

Exercise 5 (Keeping your systems secure (Bonus: 1 Points)).

Are there any new vulnerabilities for your Debian or OpenBSD system since last week (29.04.2016 at 23.59)? If so: state one, name the programming mistake, decide if you are affected or not, and report if there are any known work-arounds or patches.

Not solved