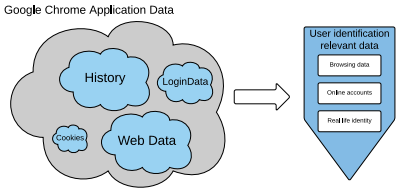
BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2016
http://excel.fit.vutbr.cz

Identification of specific Google Chrome user based on analysis of its application data

Daniel Dušek*



Abstract

This paper is primarily focused on data structures — and their meaning — within *Google Chrome application* which are relevant to identification of a specific user that was using the browser. In order to gain access to and interpret relevant data, I extracted full contents of some of the Google Chrome *sqlite3* databases that store information about user. I empirically determined meanings of certain values stored within *History* database and reverse-engineered meanings for values contained in other databases such as *Login Data*, *Web Data* or *Cookies*. To demonstrate and simplify the process of displaying extracted data I developed application *SD4Gen* that produces fingerprint-like report with the best-effort user identifying output and number of complete lists of user related data. The reader can benefit from information, conclusions and results of this work in a way to understand structure of stored data in *Google Chrome databases* as well as its internal representation in *Google Chrome application*. Conclusions and results are described in a way that they can be used as a base to reader's application implementation or just to help reader understand what plain values in database represent.

Keywords: Computer Forensics — Google Chrome — Browsing History — User Identification

Supplementary Material: Downloadable research results

*xdusek21@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

With growing importance and omnipresence of computers in today's World also grows the danger of computer abuse. One of the possible ways to abuse computer and potentially commit a crime is using the Internet and Internet browser. Given such situation it is important to be able to identify the person responsible for the abuse. Sometimes securing the attacker's computer itself will not be enough to identify the actual attacker. On occasions like the one described, understanding the internal representation of user-identifying data within the browser is needed. This paper deals with understanding the structure and meaning of user-identifying data within *Google Chrome* browser.

Most of user data in *Google Chrome* is stored in *sqlite3* databases. Such databases contain loads of traces of a specific user that used the browser. Unfortunately they also contain data that is not relevant — a lot of it. One of the core objectives of this work is to narrow down the list of places where we should look at when it comes to identifying the user. Next objective, as important as the previous one, is to explain what actual values stored in databases mean.

Ideal solution should aim for identification of the tiniest possible set of tables and columns that lead to identification of specific user of *Google Chrome* browser.

Most of values stored in databases are in integer data type, ideal solution should explain meaning of those values and interpret them in human readable format.

At the present time, there are no known solutions focused on the same topic as this paper. However there are notable on-line sources. Broadly conceived Forensics Wiki[1] offers whole section for *Google Chrome* that contains mainly information about files' locations and handy scripts for conversion of mixed time stamps used by the browser. More specialized *Google Chrome Extension Documentation*[2] focuses on interpretation of values stored in database in a human readable format, but without a connection to real values stored in databases. None of the mentioned sources is specifically targeted to identify specific user and none of the sources provides information about real values that are stored in the database.

In proposed solution there are mainly two approaches combined — empirical and reverse-engineering one. Empirical approach is applied on easily forge-able data such are *visit transitions* or *download interrupt reasons*, while reverse-engineering approach over *Google Chromium* source code is applied to more difficult tasks.

My proposed solution is in comparison to ones described above more focused on meaning of values stored in database and its relevance to identification of specific user. Other than that, my solution also targets to describe meaning of values stored in database.

2. Google Chrome Database Structure

Almost all of important data relevant to user activity within *Google Chrome* are stored in *sqlite3 databases*[1]. Access to data can be gained via any *sqlite3 client*, or by proper database file parsing.

Cookies database contains cookie records that visited sites stored into browser. Names and origins of their values may play important role in user identification. Database structure can be seen in Figure 1.

Name	Object
meta	table
cookies	table
sqlite_autoindex_meta_1	index
sqlite_autoindex_cookies_1	index
domain	index
is_transient	index

Figure 1. Simplified view of structure of Google Chrome Cookies database file

History database is one of the most important when dealing with user identification. Browsing habits, searched keywords, complete list of downloaded items, all of it can be found there. Entity-Relationship diagram of History database can be seen in Figure 2.

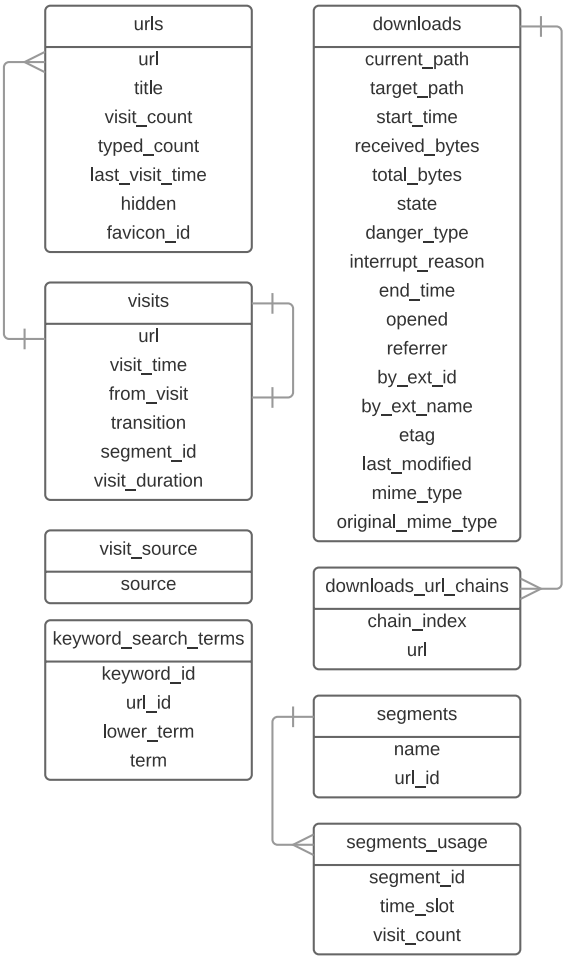


Figure 2. Entity-Relationship diagram of Google Chrome's history database, using Crow Foot Notation.

Login Data database contains table *logins* inside of which we can find saved login information used on various sites. Password is stored encrypted in `BLOB` data type. Database also contains additional information about specific login forms, sites and user's choices when saving login information.

Web Data database is holy grail of all relevant databases — all autofill values, names, addresses, phone numbers, emails and much more filled on different websites are present here. For closer look see Figure 3.

Irrelevant tables and indexes on databases are omitted in this description.

3. Empirical value meaning determination and reverse-engineering approach

To determine meaning of specific values, two approaches were combined — one based on usage of empirical methods and one based on looking into *Google Chromium*

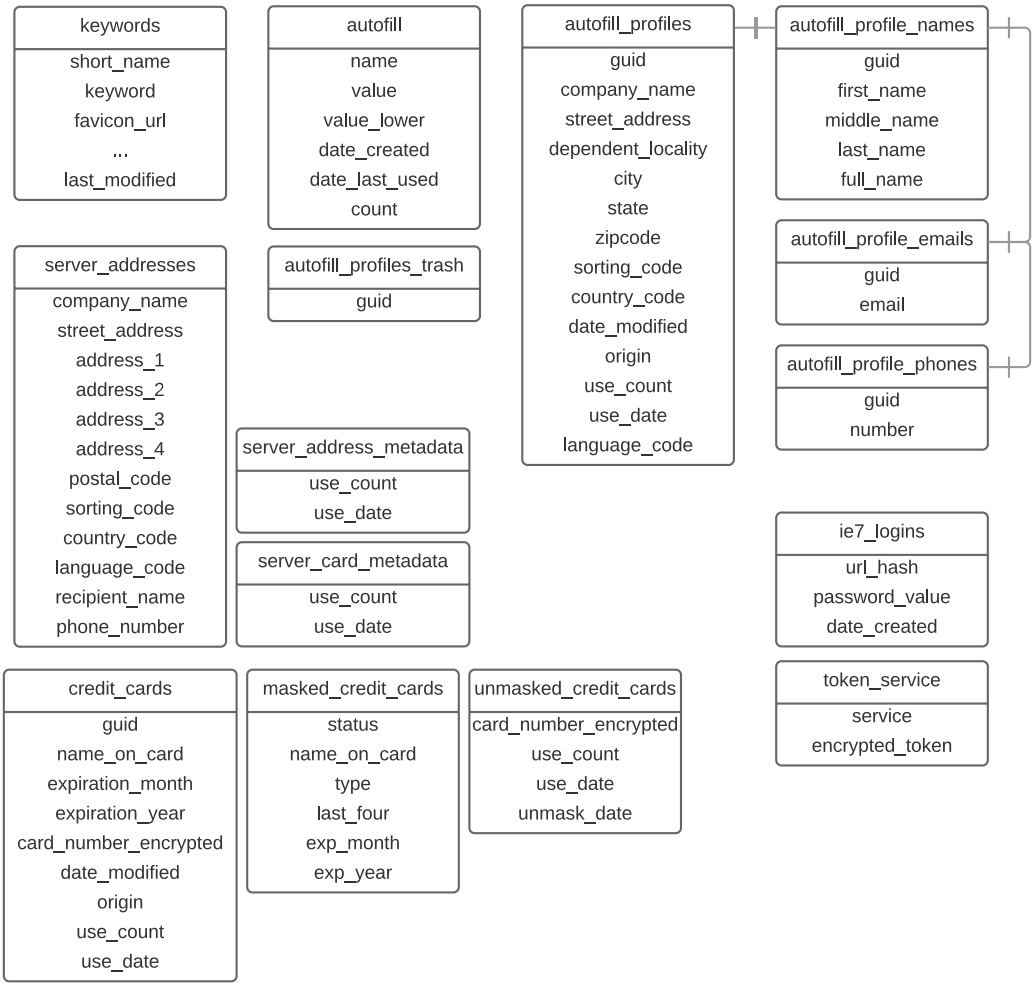


Figure 3. Entity-Relationship diagram of Google Chrome's Web Data database, Crow Foot Notation. Column names of *keywords* table were truncated. Most of the truncated values are flags of different kinds and are unimportant.

source code and determining meaning of specific values.

The **empirical approach** is based on situation simulation and evaluating results. For example, when working on *downloads* table, multiple downloads were launched and interrupted by different causes in order to determine numeric values representing download interrupt cause.

The **reverse-engineering approach** is based on gathering values from *Google Chromium* source code and their comparison with human-readable representation provided by *Google Extensions Documentation* [2]. This way I for example discovered that 64-bit integer value of *transition* column is possible to minimalize down to 8 bits by applying logical operation `AND` on it with `0xFF`, without losing meaning relevant to user identification.

Inspection of source code also revealed that there are three types of time stamp used across databases:

1. **CTime format**, represented by `VARCHAR` data type — used mostly in *last_modified*, *last_used* or *last_access* columns across the databases,
2. **Epoch time**, represented by `INT` data type — used for time stamps in Web Data database,
3. **windows FILETIME**, represented by `INT` data type — stored on 64 bits, used across all databases for time stamps of various kind.

4. Narrowing down identification-relevant tables and columns

This section covers selection of data relevant to specific user identification and reasoning behind excluding certain tables and columns from the final set of relevant data. This also briefly covers *SD4Gen* application that was implemented to demonstrate that final set of tables and columns can be used to help in identification of specific user.

4.1 Excluded and relevant data selection

The process of narrowing down identification-relevant information was realized by iterative manual reviewing of values stored in tables of each database and

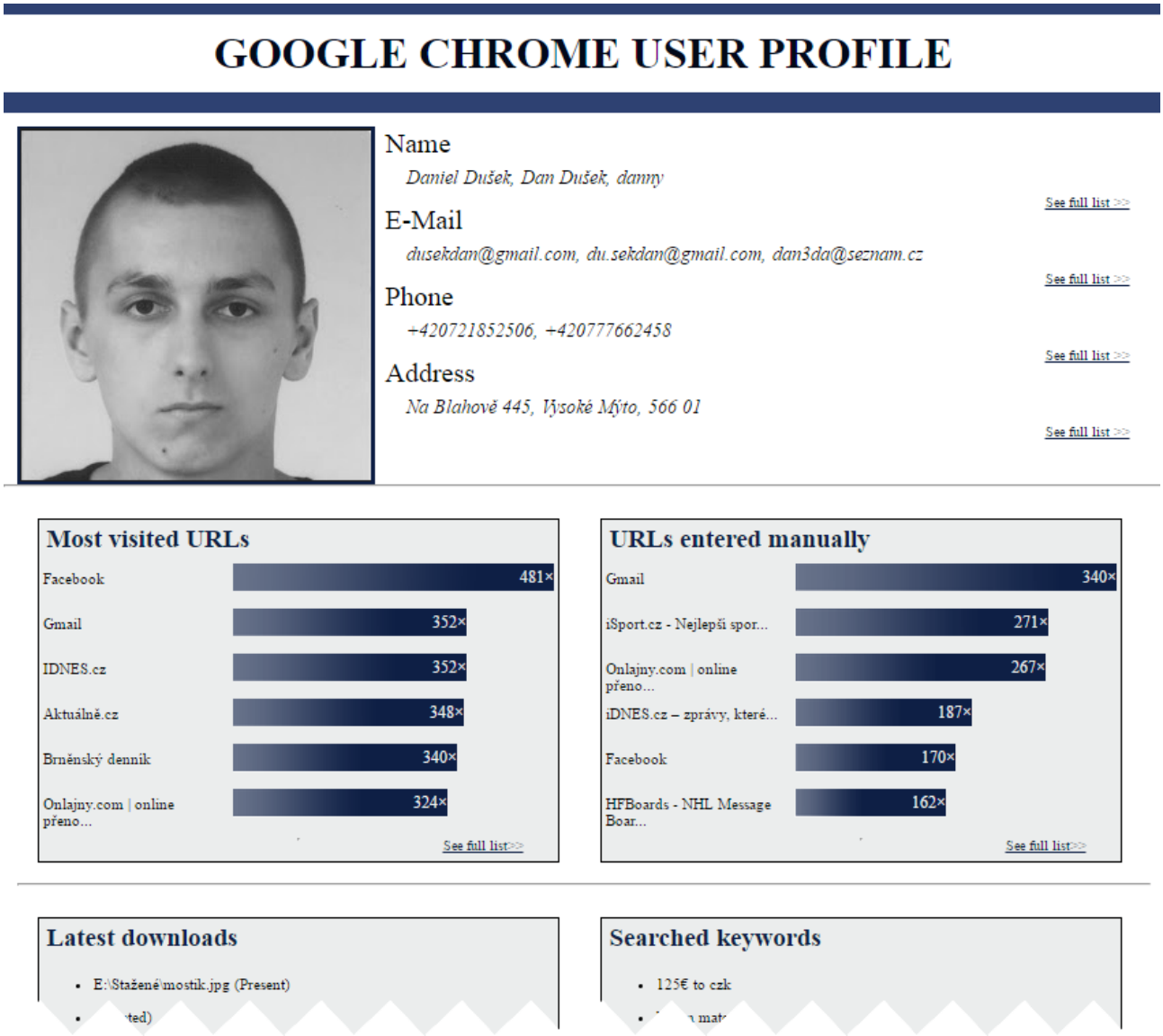


Figure 4. Screenshot of output generated by application SD4Gen demonstrating extraction of relevant information leading to identification of specific user based on analysis of Google Chrome's application data. In this case, SD4Gen application identified the author of this paper based on set of tables and columns based on his browsing habits — using final set of tables and columns presented in Section 5.

evaluating whether the information may potentially lead to identification of specific user or not. In process of determination whether to take information into account or not, the information was validated with findings from A. Baker's article [3] and from Bechar-Israeli's [4] article. Information determined by mentioned authors as relevant were taken into account and are present in final set of table and columns.

Generally all tables and columns that serve to internal purposes of *Google Chrome* application were excluded from the final set of tables and columns. Same rule applies for redundant or derivable information.

As desirable or relevant were on the other hand marked tables and columns containing sensitive user data such are user-names, mails, addresses, phone numbers and similar.

Demonstration of final set of tables and columns is present in section 5.

4.2 Creation of user fingerprint

There are several main items that are important to pay attention to, when it comes to identification of specific user. Most important traces that user leaves in his browser are his name, email, phone and address. Such data can be accessed via *Web Data* databases, and that is the way application *SD4Gen* gets its data for the resulting fingerprint.

Several issues with gathering data from *Web Data* database exists. For example, when the user uses multiple usernames, it is an issue to find out which one is the most identifying one. Application *SD4Gen* uses algorithm to find the most relevant one, nature of the al-

gorithm is described bellow.

Another issue that was encountered is a fact that multiple users may use the browser. In that case, application includes all profiles to final fingerprint and presents the most used/most active user profile as the most relevant identified user — there is often the need for linking real user identity to her/his online identity.

The most relevant information determination is based on multiple algorithms that evaluate whether the information is relevant or not. Algorithms are usually considering the number of uses, corresponding format (e.g., real name should have at least one space contained), proper context (e.g., real name will not be probably contained within email labeled input element), database table it is stored in, and similar.

Resulting finger print is displayed as showcase of most relevant user information with link to all other information that user left in the browser. To this showcase is included: real name, username, email, telephone, address and then sets of most visited pages, hand-typed pages, latest user downloads and searched keywords. Former five pieces of information were selected because they are tightly linked to user identity and the latter four were selected because it is possible to learn a lot about user's personality, or in case of searched keywords, about his intentions [5].

4.3 Displaying relevant data and best-effort user identity with SD4Gen

Application *SD4Gen* that was implemented to demonstrate usefulness of final set of tables and columns is capable of extracting given information from sqlite3 database and interpreting them in human readable format. See Figure 4 for example of output produced by SD4Gen application.

5. Reduced set of tables and columns relevant to user identification and selected columns values' meaning

This section brings demonstration of selected columns values meanings determined by my research. Furthermore this section presents final complete set of concrete tables and columns that were determined to be relevant to specific user identification.

5.1 Selected columns values' meaning

Meaning of specific values stored in selected columns are represented in form of tables. See tables Table 1, Table 2, Table 3 and Table 4.

Downloads.danger_type column values	
Column value	Description
0	Not dangerous
1	File dangerous to system (*.pdf)
2	File URLs is malicious
3	File content is malicious
4	Potentially dangerous (*.exe)
5	Unknown/Uncommon content
6	Dangerous, used ignore it
7	File from dangerous host
8	Dangerous to browser
9	Maximum (internal)

Table 1. Table explains meaning of values stored in column danger_type in downloads table in Google Chrome History database file.

Visits.transition column values (mask 0xFF applied)	
Column value	Description
0	Link
1	Typed
2	Bookmark
3	Auto subframe
4	Manual subframe
5	Generated
6	Start page
7	Form submit
8	Reload
9	Keyword
10	Keyword generated
Other	Not recorded

Table 2. Table explains meaning of values stored in column transition in visits table in Google Chrome History database file. It is necessary for 0xFF bit mask to be applied on column values before comparing.

Downloads.state column values	
Column value	Description
0	Download in progress
1	Download completed
2	Download cancelled
3	Download interrupted
4	Maximum download state reached

Table 3. Table explains meaning of values stored in column state in downloads table in Google Chrome History database file.

5.2 Reduced set of tables and columns from analyzed databases

In following paragraphs, special formatting is used. For each database there is an enumeration of its tables with count of table's column in brackets. Bold table name indicates that columns of given table are present in reduced set of tables and columns. When there are two numbers in brackets, the former represent total col-

Downloads.interrupt_reason column values	
Column value	Description
0	Not interrupted
1	Generic file operation failure
2	Access to file denied
3	Not enough free space
5	File name is too long
6	File is too large for file system
7	Virus infected file
10	File in use (transient error)
11	File blocked by local policy
12	File security check failed
13	Download revive error
20	Generic network failure
21	Network operation timed out
22	Connection lost
23	Server has gone offline
24	Network request invalid
30	Server operation failed
31	Unsupported range request
32	Request does not meet precondition
33	Server does not have requested data
40	User cancelled download
41	User shut down the browser
50	Browser crashed (internal only)

Table 4. Table explains meaning of values stored in column interrupt_reason in downloads table in Google Chrome History database file.

umn count within table and the latter represent number of columns in final reduced set.

Database Cookies contains 2 tables — *meta(2)*, ***cookies****(14,7)*. To final set is included only one table, with 7 relevant columns.

Database History contains 10 tables — *meta(2)*, ***URLs****(8,5)*, ***visits****(7,4)*, *visit_source(2)*, ***keyword_search_terms****(4,2)*, ***downloads****(18,10)*, ***downloads_url_chains****(3,3)*, *segments*, *segment_usage*, *sqlite_stat1*. In final set there are included total of 5 tables and 24 columns from History database relevant to the topic.

Database Login Data contains 3 tables — *meta(2)*, ***logins****(23, 10)*, ***stats(4)***. In final set, one table and four columns from Login Data database are included.

Database Web Data contains 16 tables — *meta(2)*, ***autofill****(6,5)*, *server_card_metadata(3)*, *server_addresses(13)*, *server_address_metadata(3)*, *keywords(25)*, *ie7_logins(3)*, *token_service(2)*, *credit_cards(9)*, ***autofill_profiles****(14,12)*, ***autofill_profile_names****(5,4)*, ***autofill_profile_emails****(2,1)*, ***autofill_profile_phones****(2,1)*, *autofill_profiles_trash(1)*, *masked_credit_cards(7)*, *unmasked_credit_cards(5)*. In final set, 5 tables with 25 relevant columns are included from Web Data database.

6. Conclusions

This paper discussed *Google Chrome* internal database structure and meaning of the data contained in terms of identification the specific user. To determine meaning of data in database two approaches were used — empirical and reverse-engineering one. Final set of tables and columns relevant to user identification was created in the process and is accessible via supplementary materials to this paper.

Total number of tables and columns present in 4 analyzed databases is 31 tables and 201 columns. I have been able to reduce the number of places to look at by roughly 67% to 12 tables and 68 columns inside the tables, by excluding not relevant and/or derivable tables and columns. Decreased number of tables and columns to inspect when trying to identify the user can lead to faster identification and potentially faster culprit capture.

This work significantly reduces amount of data required to identify specific user based on his browsing activity and habits in *Google Chrome* browser. Consequently paper explains meaning of internal values stored within certain database columns and can serve as a handbook when working with those columns.

Outcome of the *SD4Gen application* was tested on multiple users — including author of this paper, David Riha, Tomas Kaplan and two others that wished to remain anonymous. In four of five cases, resulting report lead to identification of specific user. Since the application is of privacy invading nature, I was not able to perform testing on more subjects, but I intend to do more testing in the future.

Anyone interested in user identification based on traces the user leaves in *Google Chrome* browser can benefit from results of this paper. Reader can learn about structure of Cookies, History, Login Data and Web Data databases and meaning of certain values that *Google Chrome* uses internally. I am currently working on SD4Gen application for my bachelors thesis and I intend to use conclusions and results I have researched to optimize my application.

Acknowledgements

I would like to thank my supervisor Ing. Pavel Očenášek, Ph.D for his assistance and guidance in process of composing my bachelors thesis from which this paper origins.

References

[1] Joachim Metz. Google chrome - forensicswiki, October 2015.

[2] Google chrome extensions. https://developer.chrome.com/extensions.

[3] Adam Baker. Top 16 pieces of your information identity thieves crave. http://www.manvsdebt.com/top-16-pieces-of-your-information.identity-thieves-crave/, August 2009.

[4] Haya Bechar-Israeli. From bonehead to clonehead: Nicknames, play, and identity on internet relay chat. https://www.mediensprache.net/archiv/pubs/2035.html, August 2009.

[5] Eoghan Casey. *Digital Evidence and Computer Crime: Forensic Science, Computers, and the Internet, Second Edition*. Academic Press, 2 edition, 2004.