











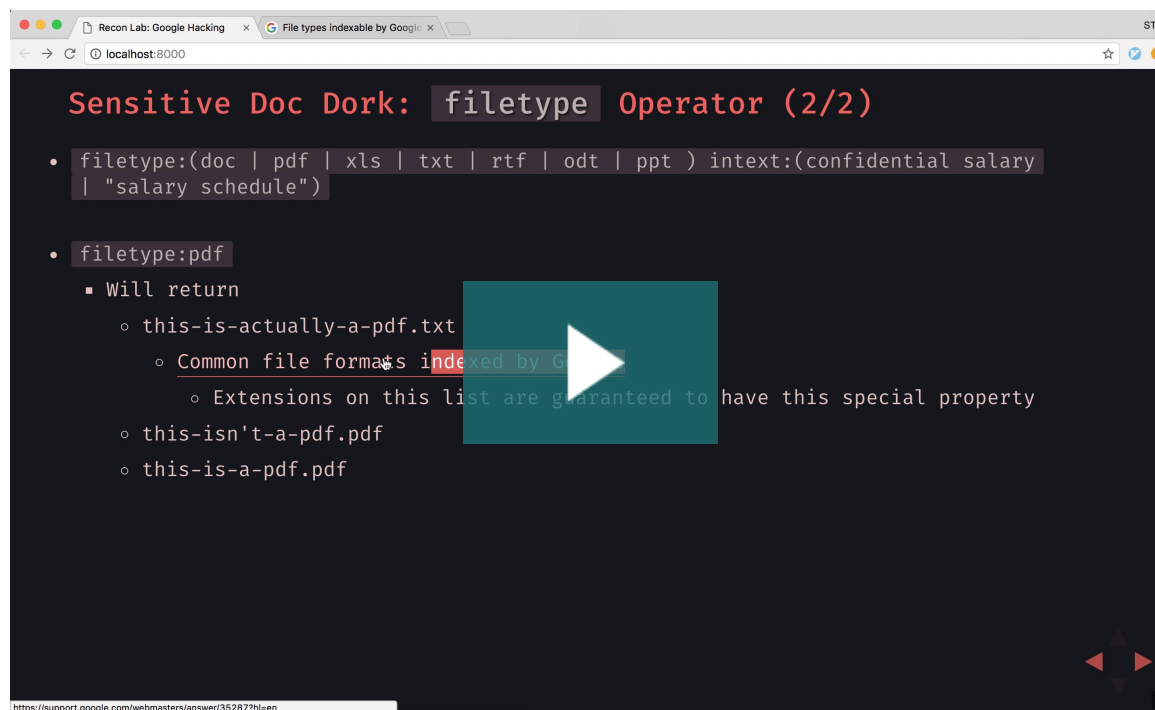


Ep. 1 - Ethical Google Hacking

Google Hacking

-   Ethical Google Hacking - Intro. (Part 1) (7:50)
-   Ethical Google Hacking - Sensitive Doc Dork (Part 2) (13:11)
-   Ethical Google Hacking - Proxy Log Dork (Part 3) (9:26)
-   Ethical Google Hacking - Error Log Dork (Part 4) (3:15)
-   Ethical Google Hacking - Admin Functionality Dork (Part 5) (2:42)
-   Ethical Google Hacking - Further Learning (Part 6) (4:48)

Ethical Google Hacking - Sensitive Doc Dork (Part 2)



Errata: stools.tools/ethical-google-hacking-1-errata



Intro: What Is Google Hacking?

- Leverage the power of Google for recon through advanced search operators
- Pioneered by Johnny Long
 - [Google Hacking for Penetration Testers, Third Edition](#)
 - [Google Hacking Database](#)
 - Search functionality needs improvement
 - Good for viewing categories
- <https://github.com/JohnTroony/Google-dorks/blob/master/google-dorks.txt>
 - Good for overall search
 - After we learn a search operator, search here for other applications

Intro: General Info.

- Level: All
- Course [Notes](#) and [Errata](#)
 - Focus on search operators in practical situations
 - Not every Google search operator will be covered
- Examples
 - Phishing campaign simulation
 - Leveraging financial documents from Board Of Director's meetings
 - Searching nginx logs for error responses while adjusting for output variability

- How to find admin area source code (or other functionality)
- Search timestamp ranges within PHP error logs

Intro: But Why?

- From a recon perspective, why is Google hacking advantageous?
 - Passive
 - Hard to trace
 - Google makes the connection to the site, not you

Intro: Additional Help

- Search for `Google Dorking`
 - Synonymous with Google Hacking
- Helpful websites
 - <http://www.googleguide.com/contents/>
 - https://www.google.com/advanced_search
 - Not as powerful
 - Can leverage if you need to get going ASAP

Sensitive Doc Dork: Background

- Scenario
 - Phishing campaign will be aimed at

- State employees who recently got a raise through the budgeting process
- “confidential employees”
 - State employees who have access to privileged information
- Phish email from “HR”
 - “Due to your recent salary adjustment, we need to confirm your banking information. Click here to confirm your bank account on file”
 - Link will redirect to a fake employee portal that steals login credentials
 - Social Engineering
 - Trust is implicitly built through disclosure of sensitive information
 - This information is commonly found via Google Dorks

Sensitive Doc Dork: Logical Operators

- `filetype:(doc | pdf | xls | txt | rtf | odt | ppt) intext:(confidential salary | "salary schedule")`
 - `()`
 - Logical grouping
 - `OR`
 - Note the uppercase
 - AKA `|`
 - If there isn't an explicit `|`, an `AND` is implied

- Within text
 - Googling `WPA2 KRACK Vulnerability`
- Adjacent search operators

Sensitive Doc Dork: `filetype` Operator (1/2)

- `filetype:(doc | pdf | xls | txt | rtf | odt | ppt) intext:(confidential salary | "salary schedule")`
- NOT
 - `filetype: (doc | pdf | xls | txt | rtf | odt | ppt)`
 - True for all operators
- Common file formats indexed by Google
 - Can search for file extensions not on this list
 - Ex: `filetype:md`
 - Q: Why would this be useful?

Sensitive Doc Dork: `filetype` Operator (2/2)

- `filetype:(doc | pdf | xls | txt | rtf | odt | ppt) intext:(confidential salary | "salary schedule")`

- `filetype:pdf`
 - Will return
 - `this-is-actually-a-pdf.txt`
 - **Common file formats indexed by Google**
 - Extensions on this list are guaranteed to have this special property
 - `this-isn't-a-pdf.pdf`
 - `this-is-a-pdf.pdf`

Sensitive Doc Dork: `intext` Operator

- `filetype:(doc | pdf | xls | txt | rtf | odt | ppt) intext:(confidential salary | "salary schedule")`
- Helpful for constraining a search to a document's body
 - Regular Google search can match page titles, items in the url path, etc.
- `intext:(confidential salary | "salary schedule")`
 - Has `confidential` AND `salary` somewhere in the text body
 - `"salary schedule"`
 - Exact match
 - We leave this search relatively vague to capture other results of interest
 - We don't do `intext:("confidential employee" | "salary schedule"`

- Problems
 - Look at query
 - Too many false positives

Sensitive Doc Dork: `inurl` Operator

- `filetype:(doc | pdf | xls | txt | rtf | odt | ppt) intext:(confidential salary | "salary schedule") inurl:(confidential "board approved */*/17")`
- `confidential`
 - Must be in the url
- `"board approved */*/17"`
 - Can be in the url or anywhere within the document
 - `*` is expanded to one or more words
 - Cross-check via bold in the results output
 - Search Ex.
- Doesn't work well for non-words
 - Ex: `inurl:"sid=*"`
 - `sid` is for a PHP session
 - Through a proxy log dork, we can find sensitive urls/url parameters

Proxy Log Dork: Why Search Through Proxy Logs?

1. URL Data leakage

- Common for all GET requests/responses to be logged at the proxy layer
- Best practice is to place sensitive information within the POST body
 - Ex: session id, sensitive tokens, SSNs, etc.
 - Often placed within a GET request
 - Ex: <https://example.com/256993ac-ba65-11e7-8e6d-0242ac110003/profile>
 - Ex: <https://example.com/profile?sid=256993ac-ba65-11e7-8e6d-0242ac110003>

2. Abnormal response codes

- https://en.wikipedia.org/wiki/List_of_HTTP_status_codes
- Example in next slide

Proxy Log Dork: **AROUND (X)** Operator

- **TERM 1** **AROUND (2)** **TERM 2**
 - **TERM 1** is within 2 words of **TERM 2**
 - Capitalize **AROUND** for more consistent results
- Useful for searching documents where the ordering of terms can be customized

- Ex: Logs
- Nginx Log Ex:
 - `- - - - "GET / HTTP/1.1" STATUS_CODE - - -`
 - Put `-` to simplify
- `filetype:log inurl:(access.log | error.log) intext:("HTTP/"`
`AROUND(5) 500) -site:github.com`

Proxy Log Dork: `site` Operator

- Scopes a search to a particular domain
 - Can even be a TLD
 - `site:.net`
- `site:github.com`
 - Will match `github.com` and `*.github.com`
 - Leave out `www` to ensure search of all subdomains
- `filetype:log inurl:(access.log | error.log) intext:("HTTP/"`
`AROUND(5) 500) -site:github.com`
 - `-site:github.com`
 - Helps us remove example logs that are false positives
 - Or are they?
 - For targeted search don't discard

- Stackoverflow for recon

- Review Ex.

– Operator

- `-site:github.com -next -last -reply -"I want to leave this out"`
- Make sure search results don't contain a given...
 - operator
 - word
 - `-next` will help negate help forum results
 - phrase

Error Log Dork: `X..Y` (Range) Operator

- Finds a given number range
- `warning error on line php filetype:log 2015..2017`
 - Search php error logs for a given timestamp
 - Ex.

Admin Functionality Dork: `inanchor` Operator

- Finds text within a link/anchor
- `inanchor:admin site:hackthissite.org`
 - Great way to find admin portals
- How can we remove some of the clutter from the results?
- Ex.

Admin Functionality Dork: `intitle` Operator

- Searches through page titles
- `inanchor:admin site:hackthissite.org -intitle:"view topic"`
- Why did the source code come up in the results?
- Ex.

Further Learning

- Overall Google functionality
 - <http://www.googleguide.com/contents/>
- Operators
 - <https://support.google.com/websearch/answer/2466433?hl=en>
 - `cache:`

- Tools
 - <https://github.com/Ekultek/Zeus-Scanner>
- Dork lists
 - Google Hacking Database
 - <https://github.com/JohnTroony/Google-dorks/blob/master/google-dorks.txt>