Looking for Honey Once Again: Detecting RDP and SMB Honeypots on the Internet

Fabian Franzen, Lion Steger, Patrick Sattler, Johannes Zirngibl
Technical University of Munich

6th June 2022, WTMC'22

Introduction

What are Honeypots?

- Mimic vulnerable service, learn something about the attacker
- Low-Interaction: Simple implementation, easy deployment & maintenance, only basic functionality
- High-Interaction: Mimic service as complete as possible

Honeypots

What are Honeypots?

- Mimic vulnerable service, learn something about the attacker
- Low-Interaction: Simple implementation, easy deployment & maintenance, only basic functionality
- High-Interaction: Mimic service as complete as possible

Why should we look for them?

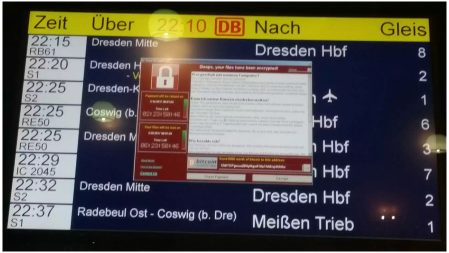
- Attacker will usually avoid them…
  - Therefore, we should also know how to detect them
- Censys.io and Shodan.io tag their search results with honeypot labels

Introduction

Technische Universität München

Why SMB & RDP?

- Very common protocols in the Windows world
  - SMB: Windows RPC and File-Exchange Protocol
  - RDP: Remote Access to Windows UI
- Subject to remotely exploitable bugs in the past
  - EternalBlue (CVE-2017-0144)
  - BlueKeep (CVE-2019-0708)
- Gap in literature: HTTP, SMTP, SSH, Telnet and ICS Honeypots have been in focus
- **How many honeypots are deployed in the Internet?**



Picture: Martin Wiesner / heise.de

Creating a Honeypot Detector

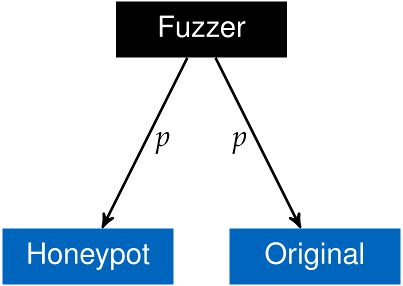
Technische Universität München

Mission Statement

- How good can open-source honeypots for RDP and SMB be fingerprinted?
  - Analyze the existing implementation, create fingerprints
- How many of these honeypots are deployed on the Internet?
  - Derive a scanner from the fingerprints, conduct an internet-wide scan
- Does it matter? Do attackers react on the presence of honeypots?
  - Deploy own honeypots and benign machines
  - Check the recorded traffic for different attack patterns

Fingerprinting Algorithm

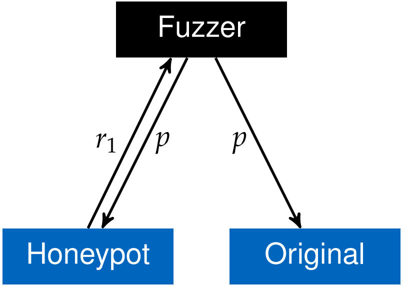
1. Analyze protocol
2. Implement a basic client implementation
3. Add a custom fuzzer to do differential fuzzing
   3.1 Send same probe p to honeypot and benign implementation



Creating a Honeypot Detector

Fingerprinting Algorithm

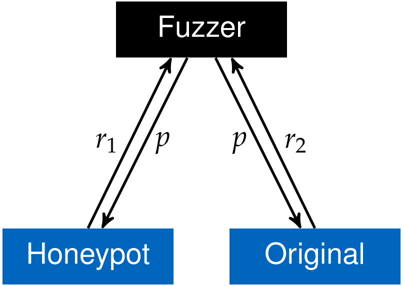
1. Analyze protocol
2. Implement a basic client implementation
3. Add a custom fuzzer to do differential fuzzing
   3.1 Send same probe p to honeypot and benign implementation



Creating a Honeypot Detector

Fingerprinting Algorithm

1. Analyze protocol
2. Implement a basic client implementation
3. Add a custom fuzzer to do differential fuzzing
   3.1 Send same probe p to honeypot and benign implementation

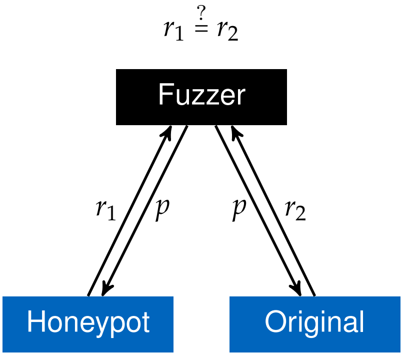


Creating a Honeypot Detector

Technische Universität München

Fingerprinting Algorithm

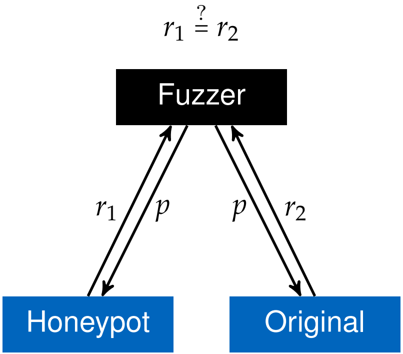
1. Analyze protocol
2. Implement a basic client implementation
3. Add a custom fuzzer to do differential fuzzing
   3.1 Send same probe p to honeypot and benign implementation
   3.2 Withdraw response r if responses $r_1 = r_2$, save p as distinctive probe otherwise



Creating a Honeypot Detector

Fingerprinting Algorithm

1. Analyze protocol
2. Implement a basic client implementation
3. Add a custom fuzzer to do differential fuzzing
   3.1 Send same probe p to honeypot and benign implementation
   3.2 Withdraw response r if responses $r_1 = r_2$, save p as distinctive probe otherwise
4. Repeat with all implementations of interest
5. Analyze distinctive packets

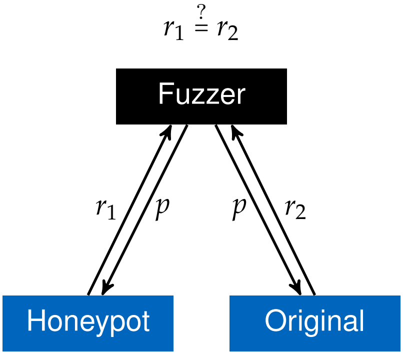


Creating a Honeypot Detector

Technische Universität München

Implementation Details

- p, r_1, r_2 may contain timestamps, IDs, random numbers → Ignore them during comparison
- A single message exchange is usually not distinctive enough! → Use a set of requests, send follow up requests
- We used different fuzzing strategies:
  - Bit-Flipping
  - Grammar based: Use plausible values
  - Both protocols are complex → Enough potential for implementation differences



Complex protocols? It could not be that bad...

Pros and Cons for honeypot implementors:

- 👍 Specification available! RDP and SMB are part of the MS Open Specification program!

Complex protocols? It could not be that bad...

Technische Universität München

Pros and Cons for honeypot implementors:

- 👍 Specification available! RDP and SMB are part of the MS Open Specification program!

- 👎 Both protocols exist since the Windows 2000/NT days...
  - SMB 1.0 was designed in early 1983 with NetBios support!
  - A lot of legacy modes that need to be supported!

Complex protocols? It could not be that bad...

Technische Universität München

Pros and Cons for honeypot implementors:

- 👍 Specification available! RDP and SMB are part of the MS Open Specification program!
- 👎 Both protocols exist since the Windows 2000/NT days…
  - SMB 1.0 was designed in early 1983 with NetBios support!
  - A lot of legacy modes that need to be supported!
- 👎 Mature protocols with a rich feature set!
  - If a specific feature combination is unsupported this yields a fingerprint!

Complex protocols? It could not be that bad...

Technische Universität München

Pros and Cons for honeypot implementors:

- 👍 Specification available! RDP and SMB are part of the MS Open Specification program!

- 👎 Both protocols exist since the Windows 2000/NT days...
  - SMB 1.0 was designed in early 1983 with NetBios support!
  - A lot of legacy modes that need to be supported!
- 👎 Mature protocols with a rich feature set!
  - If a specific feature combination is unsupported this yields a fingerprint!
- 👎 Strongly embedded into the Windows ecosystem.
  - MS RDP uses the S-Channel TLS implementation of Windows (not OpenSSL!)
  - MS RDP can interoperate with KERBEROS for authentication!

Complex protocols? It could not be that bad...

Technische Universität München

Pros and Cons for honeypot implementors:

- 👍 Specification available! RDP and SMB are part of the MS Open Specification program!
- 👎 Both protocols exist since the Windows 2000/NT days...
  - SMB 1.0 was designed in early 1983 with NetBios support!
  - A lot of legacy modes that need to be supported!
- 👎 Mature protocols with a rich feature set!
  - If a specific feature combination is unsupported this yields a fingerprint!
- 👎 Strongly embedded into the Windows ecosystem.
  - MS RDP uses the S-Channel TLS implementation of Windows (not OpenSSL!)
  - MS RDP can interoperate with KERBEROS for authentication!

→ **Basically impossible to reimplement everything.**

Fingerprint Example

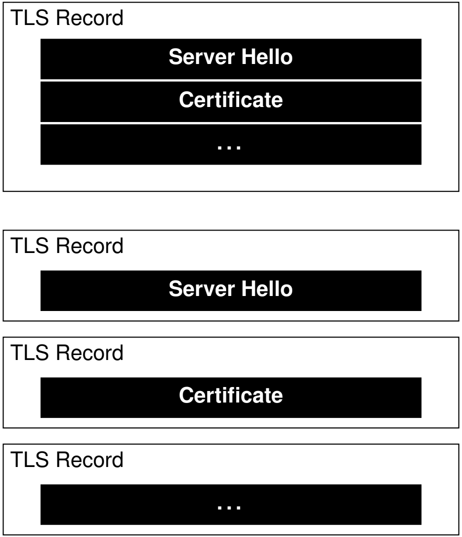
Technische Universität München

- Require exact fingerprint match
  - Filter out fields being configuration dependent
- Benign implementations answer with different capabilities or hardcoded settings
- Furthermore, they react differently to erroneous behaviour caused by our fuzzer:
  - Windows machines answer with a TCP RST
  - Error message vs no error message
  - Error ignored

Field name	XRDP	Win10	Win8	Win7	WinXP	rdpy	heralding
T.125 Conn. Resp.							
...							
Domain Parameters							
Max Channel IDs	22	✖	34	34	✖	22	✖
...							
RDP Server Data							
Server Core Data							
...							
Length	12	✖	16	12	✖	16	✖
Early Capability Fl.	✖	✖	0x1	✖	✖	0	✖
...							

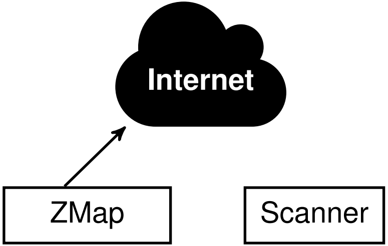
TLS-Fingerprints

- RDP uses TLS (in modern protocol versions)
- TLS offers its own surface for fingerprinting
  - Fingerprintable properties include *Cipher Suites*, *TLS Extensions*, ...
  - Tools: JA3S, JARM, ...
  - **Multiple ways to structure messages**



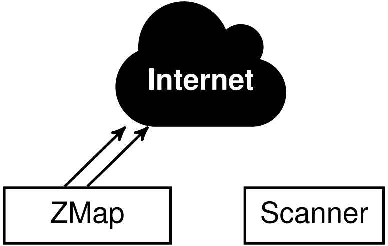
Internet Scanning

- We utilize ZMap to perform a port scan.
  - BGP dump as IP list input
- Scan only hosts that are alive on the RDP/SMB port.
- We use three probes for SMB and four probes for RDP.
  - Still allow high-scan speeds



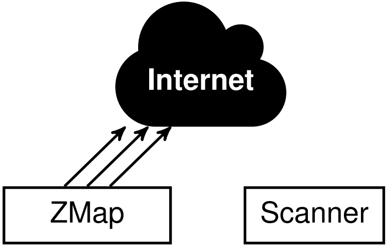
Internet Scanning

- We utilize ZMap to perform a port scan.
  - BGP dump as IP list input
- Scan only hosts that are alive on the RDP/SMB port.
- We use three probes for SMB and four probes for RDP.
  - Still allow high-scan speeds



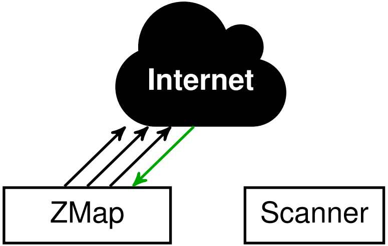
Internet Scanning

- We utilize ZMap to perform a port scan.
  - BGP dump as IP list input
- Scan only hosts that are alive on the RDP/SMB port.
- We use three probes for SMB and four probes for RDP.
  - Still allow high-scan speeds

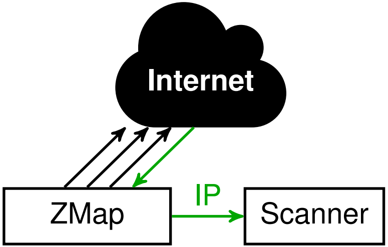


Internet Scanning

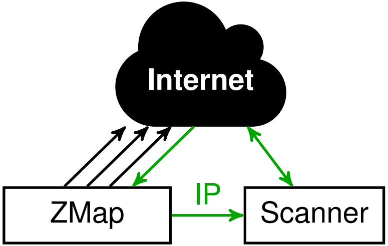
- We utilize ZMap to perform a port scan.
  - BGP dump as IP list input
- Scan only hosts that are alive on the RDP/SMB port.
- We use three probes for SMB and four probes for RDP.
  - Still allow high-scan speeds



- We utilize ZMap to perform a port scan.
  - BGP dump as IP list input
- Scan only hosts that are alive on the RDP/SMB port.
- We use three probes for SMB and four probes for RDP.
  - Still allow high-scan speeds



- We utilize ZMap to perform a port scan.
  - BGP dump as IP list input
- Scan only hosts that are alive on the RDP/SMB port.
- We use three probes for SMB and four probes for RDP.
  - Still allow high-scan speeds



Technische Universität München

Results

- 7.6 million RDP hosts and 2.7 million SMB hosts responded to ZMap.
  - This usually includes false positives, 4.2 million and 1.5 million without reaction or immediate connection close
  - 245 300 hosts on port 3359 (RDP standard port) offer a different service

Technische Universität München

Results

- 7.6 million RDP hosts and 2.7 million SMB hosts responded to ZMap.
  - This usually includes false positives, 4.2 million and 1.5 million without reaction or immediate connection close
  - 245 300 hosts on port 3359 (RDP standard port) offer a different service
- 1.9 million RDP hosts and 1.1 million SMB hosts classified as Regular Implementations

Technische Universität München

Results

- 7.6 million RDP hosts and 2.7 million SMB hosts responded to ZMap.
  - This usually includes false positives, 4.2 million and 1.5 million without reaction or immediate connection close
  - 245 300 hosts on port 3359 (RDP standard port) offer a different service
- 1.9 million RDP hosts and 1.1 million SMB hosts classified as Regular Implementations
- 1207 RDP and 1521 SMB hosts are classified as honeypots
  - attributed to well-known implementations like RDPY, DIONAEA, IMPACKET, and HERALDING

Technische Universität München

Results

- 7.6 million RDP hosts and 2.7 million SMB hosts responded to ZMap.
  - This usually includes false positives, 4.2 million and 1.5 million without reaction or immediate connection close
  - 245 300 hosts on port 3359 (RDP standard port) offer a different service
- 1.9 million RDP hosts and 1.1 million SMB hosts classified as Regular Implementations
- 1207 RDP and 1521 SMB hosts are classified as honeypots
  - attributed to well-known implementations like RDPY, DIONAEA, IMPACKET, and HERALDING
- 1 million RDP and 31 152 SMB hosts are not categorized
  - *Reminder:* We aimed for a low false-positive rate and therefore require exact fingerprint matches

Technische Universität München

Results

- 7.6 million RDP hosts and 2.7 million SMB hosts responded to ZMap.
  - This usually includes false positives, 4.2 million and 1.5 million without reaction or immediate connection close
  - 245 300 hosts on port 3359 (RDP standard port) offer a different service
- 1.9 million RDP hosts and 1.1 million SMB hosts classified as Regular Implementations
- 1207 RDP and 1521 SMB hosts are classified as honeypots
  - attributed to well-known implementations like RDPY, DIONAEA, IMPACKET, and HERALDING
- 1 million RDP and 31 152 SMB hosts are not categorized
  - *Reminder:* We aimed for a low false-positive rate and therefore require exact fingerprint matches
- 14 RDP hosts match perfectly with our RDP fingerprint except the fingerprint of the TLS stack.
  - MitM-Box? High Interaction Honeypots?

Technische Universität München

- More than 50 percent of honeypots are placed in less than 12 ASes!

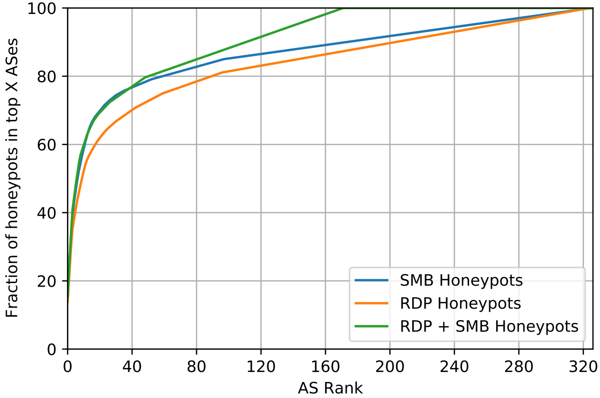


Figure: AS distribution of honeypot addresses

CO	ASN	Organization	SMB	RDP	**Total**
US	16509	AMAZON	232	167	399
US	20473	CHOOPA	126	95	221
US	14061	DIGITALOCEAN	102	90	192
DE	197540	netcup	66	72	138
TW	1659	TANet	131	1	132
US	8075	MICROSOFT	48	25	73
US	63949	Linode	33	37	70
US	14618	AMAZON	41	28	69
US	15169	GOOGLE	35	32	67
US	22773	Cox Communications	50	3	53

Table: Top 10 Autonomous systems hosting honeypots

Technische Universität München

	CO	ASN	Organization	SMB	RDP	**Total**
EC2 →	US	16509	AMAZON	232	167	399
	US	20473	CHOOPA	126	95	221
	US	14061	DIGITALOCEAN	102	90	192
	DE	197540	netcup	66	72	138
	TW	1659	TANet	131	1	132
	US	8075	MICROSOFT	48	25	73
	US	63949	Linode	33	37	70
EC2 →	US	14618	AMAZON	41	28	69
	US	15169	GOOGLE	35	32	67
	US	22773	Cox Communications	50	3	53

Table: Top 10 Autonomous systems hosting honeypots

Internet Scanning

	CO	ASN	Organization	SMB	RDP	**Total**
EC2 →	US	16509	AMAZON	232	167	399
	US	20473	CHOOPA	126	95	221
	US	14061	DIGITALOCEAN	102	90	192
	DE	197540	netcup	66	72	138
	TW	1659	TANet	131	1	132
Azure →	US	8075	MICROSOFT	48	25	73
	US	63949	Linode	33	37	70
EC2 →	US	14618	AMAZON	41	28	69
	US	15169	GOOGLE	35	32	67
	US	22773	Cox Communications	50	3	53

Table: Top 10 Autonomous systems hosting honeypots

Internet Scanning

	CO	ASN	Organization	SMB	RDP	**Total**
EC2 →	US	16509	AMAZON	232	167	399
Cloud →	US	20473	CHOOPA	126	95	221
Cloud →	US	14061	DIGITALOCEAN	102	90	192
Cloud →	DE	197540	netcup	66	72	138
	TW	1659	TANet	131	1	132
Azure →	US	8075	MICROSOFT	48	25	73
Cloud →	US	63949	Linode	33	37	70
EC2 →	US	14618	AMAZON	41	28	69
Cloud →	US	15169	GOOGLE	35	32	67
	US	22773	Cox Communications	50	3	53

Table: Top 10 Autonomous systems hosting honeypots

	CO	ASN	Organization	SMB	RDP	**Total**
	US	16509	AMAZON	232	167	399
	US	20473	CHOOPA	126	95	221
	US	14061	DIGITALOCEAN	102	90	192
	DE	197540	netcup	66	72	138
Academic →	TW	1659	TANet	131	1	132
	US	8075	MICROSOFT	48	25	73
	US	63949	Linode	33	37	70
	US	14618	AMAZON	41	28	69
	US	15169	GOOGLE	35	32	67
	US	22773	Cox Communications	50	3	53

Table: Top 10 Autonomous systems hosting honeypots

Verification

Verification of the results is hard. . . We have no ground truth!

- Some hosts have a SMB and RDP honeypot running.

Verification of the results is hard. . . We have no ground truth!

- Some hosts have a SMB and RDP honeypot running.
- We connected to a random subsample of each classification label.

Verification

Technische Universität München

Verification of the results is hard... We have no ground truth!

- Some hosts have a SMB and RDP honeypot running.
- We connected to a random subsample of each classification label.
  - For honeypots the connections have been performed by a human analyst
  - For benign hosts we used additional automated steps to confirm the low false positive rate

Verification

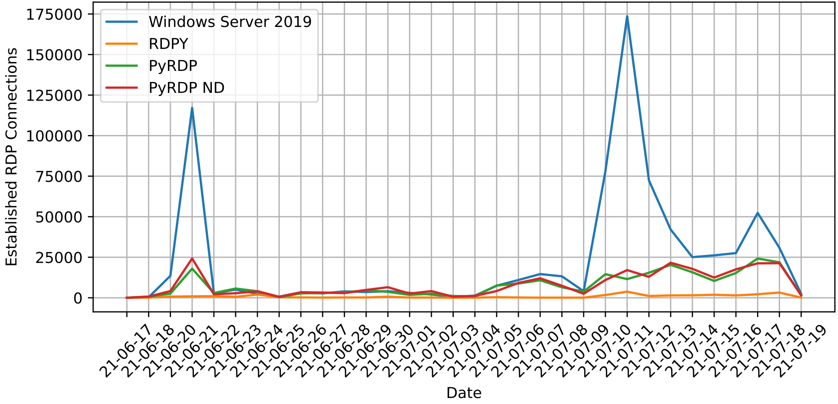
Verification of the results is hard... We have no ground truth!

- Some hosts have a SMB and RDP honeypot running.
- We connected to a random subsample of each classification label.
  - For honeypots the connections have been performed by a human analyst
  - For benign hosts we used additional automated steps to confirm the low false positive rate

- 1097 hosts have been correctly classified while only 5 have been misclassified!

Attack Analysis

Do attackers react on the presence of honeypots?
We deployed RDP honeypots and benign Windows machines for 34 days to the Internet and analyzed the results…



Attack Analysis

Technische Universität München

Observations

- We received traffic from Shodan.io, Censys.io and other not well known Internet scanning services.
- Benign hosts are preferably connected to.
- Clients connect and disconnect immediately or perform credential stuffing attacks.
- Issue: Hosts communicate! A scan of host A influences behaviour of host B.
  - i.e. Censys.io has dedicated hosts for port scanning and dedicated protocol analysis.
  - Benign hosts are prefered even if the connecting hosts has never connected to others.
  - Scans are done by Autonomous Systems / IPv4 address ranges.

Conclusion

- Low-Interaction honeypots are rarely, but still used!
- It is challenging to build a stealthy honeypot for RDP and SMB.
  - Both protocols offer a giant surface for implementation differences!
  - Differential fuzzing can be used to eliminate differences!
- We demonstrated that attacks are less common on honeypots as on benign machines in the Internet!
- Watch out for differences in your TLS implementation!

We provide code!
Check it out!

https://github.com/tum-itsec/looking-for-honey-once-again

Conclusion

Technische Universität München

- Low-Interaction honeypots are rarely, but still used!
- It is challenging to build a stealthy honeypot for RDP and SMB.
  - Both protocols offer a giant surface for implementation differences!
  - Differential fuzzing can be used to eliminate differences!
- We demonstrated that attacks are less common on honeypots as on benign machines in the Internet!
- Watch out for differences in your TLS implementation!

Thank you for listening!

We provide code!
Check it out!

https://github.com/tum-itsec/looking-for-honey-once-again