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
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
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
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?
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
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 $r_1 = r_2$, save $p$ as distinctive probe otherwise
4. Repeat with all implementations of interest
5. Analyze distinctive packets
**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

![Fingerprinting Algorithm Diagram]

Looking for Honey Once Again: Detecting RDP and SMB Honeypots on the Internet
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 $r_1$ if responses $r_1 = r_2$, save $p$ as distinctive probe otherwise
4. Repeat with all implementations of interest
5. Analyze distinctive packets
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

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
Pros and Cons for honeypot implementors:

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

- 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!
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!

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.
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!
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!
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

- 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

<table>
<thead>
<tr>
<th>Field name</th>
<th>XRDP</th>
<th>Win10</th>
<th>Win8</th>
<th>Win7</th>
<th>WinXP</th>
<th>rdpy</th>
<th>heralding</th>
</tr>
</thead>
<tbody>
<tr>
<td>T.125 Conn. Resp.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Domain Parameters</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max Channel IDs</td>
<td>22</td>
<td>✗</td>
<td>34</td>
<td>34</td>
<td>✗</td>
<td>22</td>
<td>✗</td>
</tr>
<tr>
<td>RDP Server Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Server Core Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Length</td>
<td>12</td>
<td>✗</td>
<td>16</td>
<td>12</td>
<td>✗</td>
<td>16</td>
<td>✗</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✗</td>
<td>0x1</td>
<td>✗</td>
<td>✗</td>
<td>0</td>
<td>✗</td>
</tr>
<tr>
<td>Early Capability Fl.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>✗</td>
<td>✗</td>
<td>0x1</td>
<td>✗</td>
<td>✗</td>
<td>0</td>
<td>✗</td>
</tr>
</tbody>
</table>
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
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
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
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
- 1,207 RDP and 1,521 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?
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, I_MPI, 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?
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
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
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 31152 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?
More than **50 percent** of honeypots are placed in less than **12 ASes**!

**Figure**: AS distribution of honeypot addresses
## Internet Scanning

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

**Table:** Top 10 Autonomous systems hosting honeypots
<table>
<thead>
<tr>
<th>CO</th>
<th>ASN</th>
<th>Organization</th>
<th>SMB</th>
<th>RDP</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC2</td>
<td>US</td>
<td>AMAZON</td>
<td>232</td>
<td>167</td>
<td>399</td>
</tr>
<tr>
<td>US</td>
<td>20473</td>
<td>CHOOPA</td>
<td>126</td>
<td>95</td>
<td>221</td>
</tr>
<tr>
<td>US</td>
<td>14061</td>
<td>DIGITALOCEAN</td>
<td>102</td>
<td>90</td>
<td>192</td>
</tr>
<tr>
<td>DE</td>
<td>197540</td>
<td>netcup</td>
<td>66</td>
<td>72</td>
<td>138</td>
</tr>
<tr>
<td>TW</td>
<td>1659</td>
<td>TANet</td>
<td>131</td>
<td>1</td>
<td>132</td>
</tr>
<tr>
<td>US</td>
<td>8075</td>
<td>MICROSOFT</td>
<td>48</td>
<td>25</td>
<td>73</td>
</tr>
<tr>
<td>US</td>
<td>63949</td>
<td>Linode</td>
<td>33</td>
<td>37</td>
<td>70</td>
</tr>
<tr>
<td>EC2</td>
<td>US</td>
<td>AMAZON</td>
<td>41</td>
<td>28</td>
<td>69</td>
</tr>
<tr>
<td>US</td>
<td>15169</td>
<td>GOOGLE</td>
<td>35</td>
<td>32</td>
<td>67</td>
</tr>
<tr>
<td>US</td>
<td>22773</td>
<td>Cox Communications</td>
<td>50</td>
<td>3</td>
<td>53</td>
</tr>
</tbody>
</table>

Table: Top 10 Autonomous systems hosting honeypots
## Internet Scanning

**Table:** Top 10 Autonomous systems hosting honeypots

<table>
<thead>
<tr>
<th>CO</th>
<th>ASN</th>
<th>Organization</th>
<th>SMB</th>
<th>RDP</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC2</td>
<td>US</td>
<td>AMAZON</td>
<td>232</td>
<td>167</td>
<td>399</td>
</tr>
<tr>
<td>US</td>
<td>20473</td>
<td>CHOOPA</td>
<td>126</td>
<td>95</td>
<td>221</td>
</tr>
<tr>
<td>US</td>
<td>14061</td>
<td>DIGITALOCEAN</td>
<td>102</td>
<td>90</td>
<td>192</td>
</tr>
<tr>
<td>DE</td>
<td>197540</td>
<td>netcup</td>
<td>66</td>
<td>72</td>
<td>138</td>
</tr>
<tr>
<td>TW</td>
<td>1659</td>
<td>TANet</td>
<td>131</td>
<td>1</td>
<td>132</td>
</tr>
<tr>
<td>Azure</td>
<td>US</td>
<td>MICROSOFT</td>
<td>48</td>
<td>25</td>
<td>73</td>
</tr>
<tr>
<td>US</td>
<td>63949</td>
<td>Linode</td>
<td>33</td>
<td>37</td>
<td>70</td>
</tr>
<tr>
<td>EC2</td>
<td>US</td>
<td>AMAZON</td>
<td>41</td>
<td>28</td>
<td>69</td>
</tr>
<tr>
<td>US</td>
<td>15169</td>
<td>GOOGLE</td>
<td>35</td>
<td>32</td>
<td>67</td>
</tr>
<tr>
<td>US</td>
<td>22773</td>
<td>Cox Communications</td>
<td>50</td>
<td>3</td>
<td>53</td>
</tr>
</tbody>
</table>
### Table: Top 10 Autonomous systems hosting honeypots

<table>
<thead>
<tr>
<th>CO</th>
<th>ASN</th>
<th>Organization</th>
<th>SMB</th>
<th>RDP</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC2</td>
<td>US</td>
<td>AMAZON</td>
<td>232</td>
<td>167</td>
<td>399</td>
</tr>
<tr>
<td>Cloud</td>
<td>US</td>
<td>CHOOPA</td>
<td>126</td>
<td>95</td>
<td>221</td>
</tr>
<tr>
<td>Cloud</td>
<td>US</td>
<td>DIGITALOCEAN</td>
<td>102</td>
<td>90</td>
<td>192</td>
</tr>
<tr>
<td>Cloud</td>
<td>DE</td>
<td>netcup</td>
<td>66</td>
<td>72</td>
<td>138</td>
</tr>
<tr>
<td>TW</td>
<td>1659</td>
<td>TANet</td>
<td>131</td>
<td>1</td>
<td>132</td>
</tr>
<tr>
<td>Azure</td>
<td>US</td>
<td>MICROSOFT</td>
<td>48</td>
<td>25</td>
<td>73</td>
</tr>
<tr>
<td>Cloud</td>
<td>US</td>
<td>Linode</td>
<td>33</td>
<td>37</td>
<td>70</td>
</tr>
<tr>
<td>EC2</td>
<td>US</td>
<td>AMAZON</td>
<td>41</td>
<td>28</td>
<td>69</td>
</tr>
<tr>
<td>Cloud</td>
<td>US</td>
<td>GOOGLE</td>
<td>35</td>
<td>32</td>
<td>67</td>
</tr>
<tr>
<td>US</td>
<td>22773</td>
<td>Cox Communications</td>
<td>50</td>
<td>3</td>
<td>53</td>
</tr>
</tbody>
</table>

**Internet Scanning**
## Internet Scanning

### Table: Top 10 Autonomous systems hosting honeypots

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