IDS Evasion Techniques

DefCamp 2015

Bucharest, Nov 19th-20th 2015

Tudor Damian

General Manager & IT Solutions Specialist @ Transcent

Operations Manager & Partner @ Avaelgo

Co-founder @ ITCamp & ITCamp Community

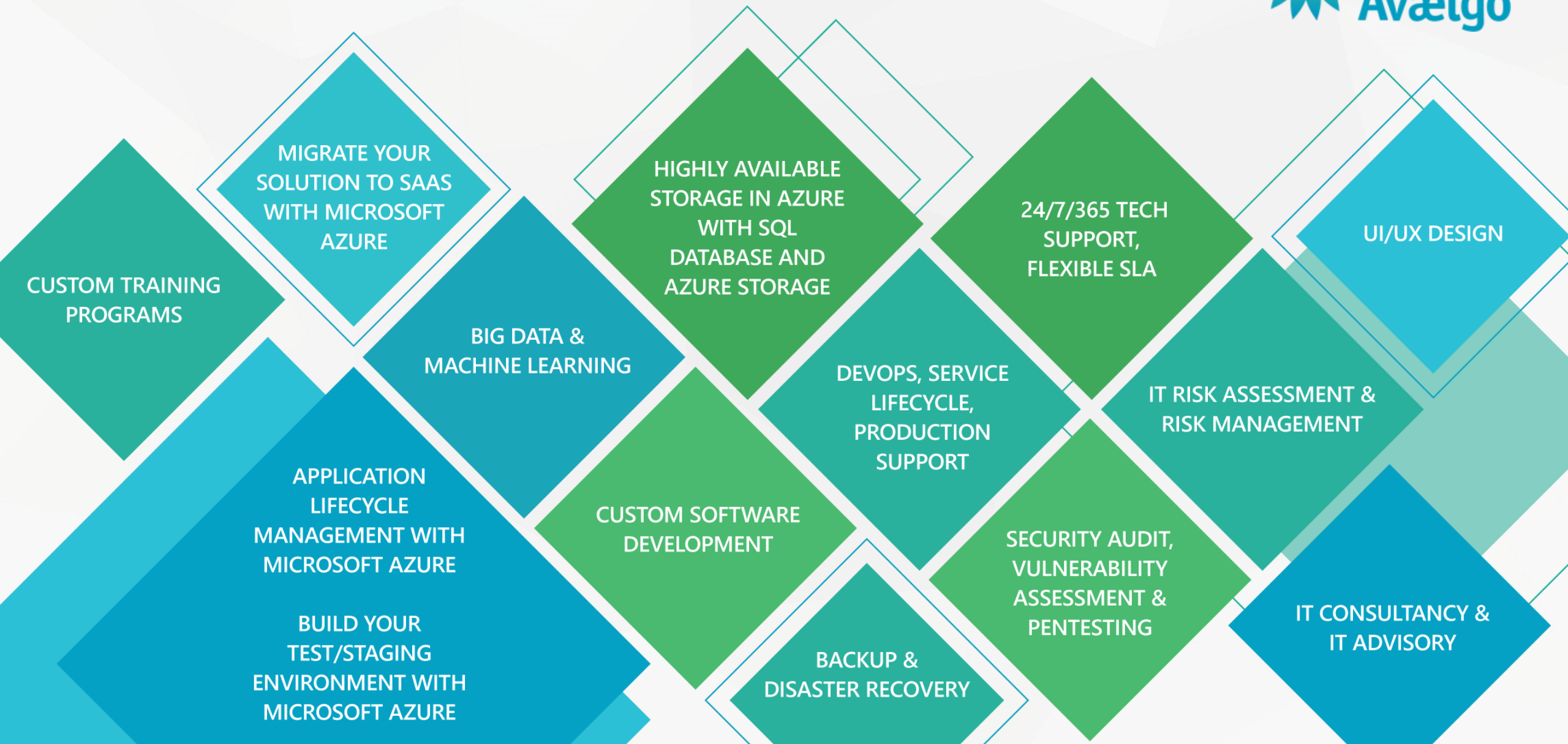
- Certified Ethical Hacker (EC-Council)
- Certified Security Professional (CQURE)
- Cloud & Datacenter Management MVP (Microsoft)

Contact: tudy@tudy.ro / www.tudy.tel

ITCAMP TRANSCENT YOUR IT DEPARTMENT Avælgo C|EH™ Certified Ethical Hacker MVP Microsoft® Most Valuable Professional

Your Technology Business Partner

Avælgo



Fair Warning

- There’s more content that we can realistically cover before lunch ☺
  - We’re only aiming for a high-level overview here, not an in-depth dive
- No need to take notes, you’ll get the slides anyway ☺

IDS Evasion Techniques

Brief Overview

- From CIDF to RFC 4765
- Physical, Network and Host IDS/IPS
- NIDS design considerations & problems

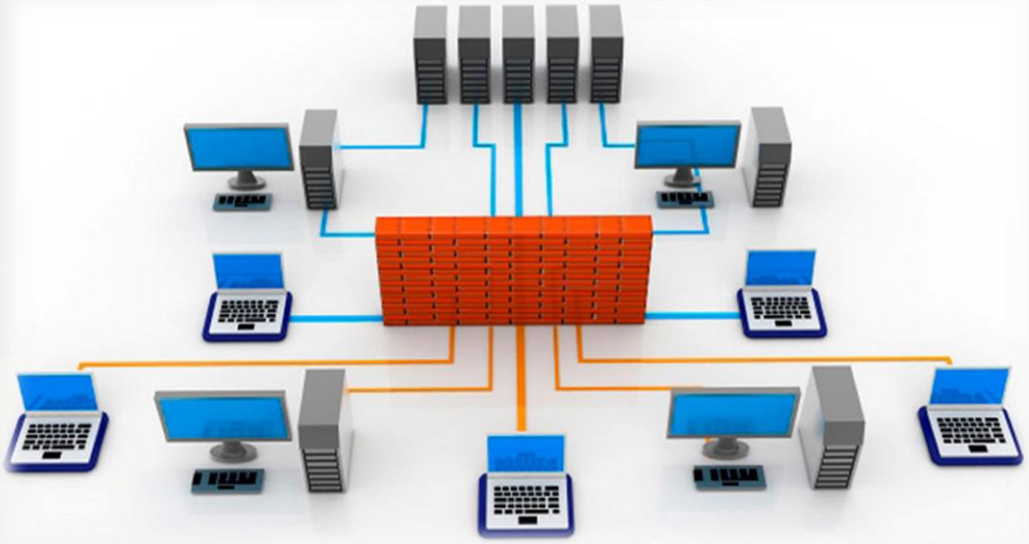
Some Advanced Evasion Techniques (AETs)

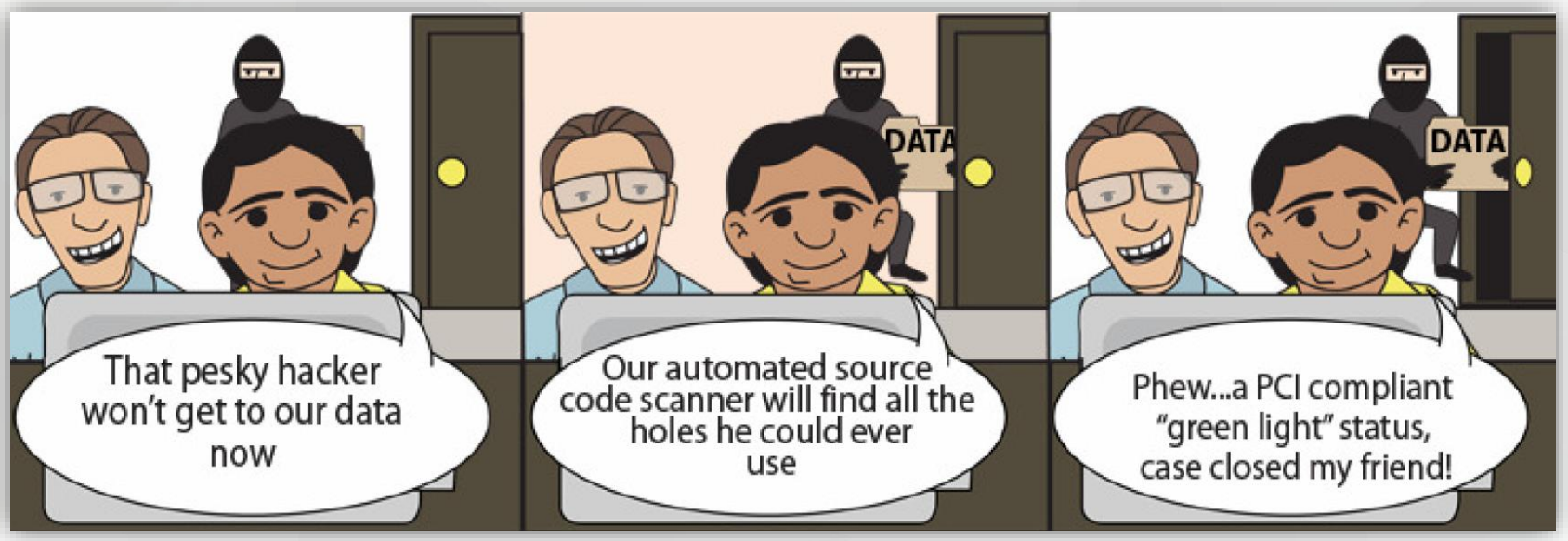
- Obfuscation (Insertion, Evasion, Session Splicing, Fragmentation)
- Denial of Service (DoS) & False Positive Generation
- Pattern-Matching Weaknesses
- Protocol Violation, TTL Attacks, Urgency Flag
- Polymorphic & ASCII Shellcode
- Encryption and Tunneling
- Application Hijacking

Potential Solutions

- Normalization
- Packet Interpretation Based on Target Host

Tools & Resources

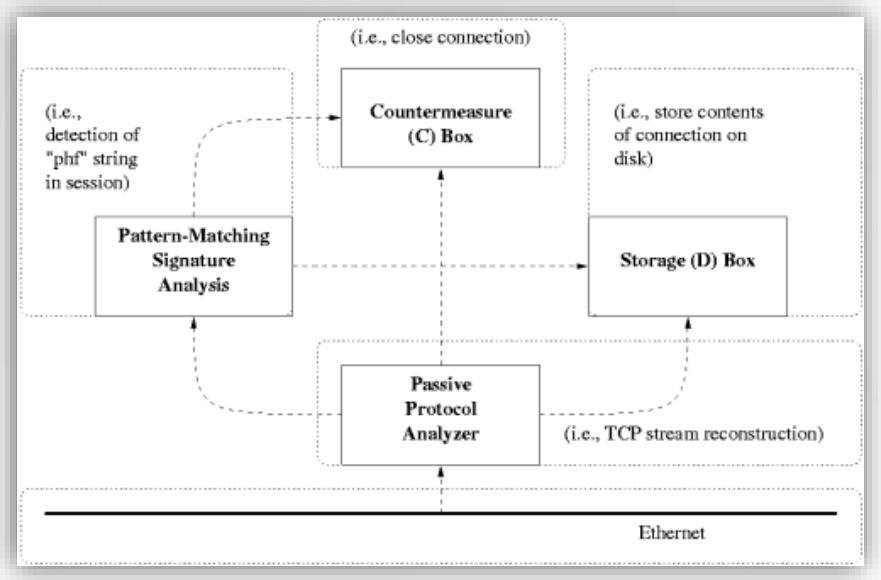




Brief Overview

From CIDF to RFC 4765

- Common Intrusion Detection Framework (CIDF)
  - An old (late 90s) attempt by DARPA (US govt's Defense Advanced Research Projects Agency) to develop an IDS interchange format
  - Started as a research project, currently dormant
- CIDF components that together define an Intrusion Detection System:
  - E-boxes - event generators (sniffers, monitors)
  - A-boxes - analysis engines (signature matchers)
  - D-boxes - storage mechanisms (loggers)
  - C-boxes - countermeasures (alarms, firewalls)
- The IETF (Internet Engineering Task Force) more recently (2007) started work on a common format (RFC 4765):
  - http://www.ietf.org/rfc/rfc4765.txt



Physical IDS

- Security Guards
- Security Cameras
- Access Control Systems (Card, Biometric)
- Firewalls
- Man Traps
- Motion Sensors

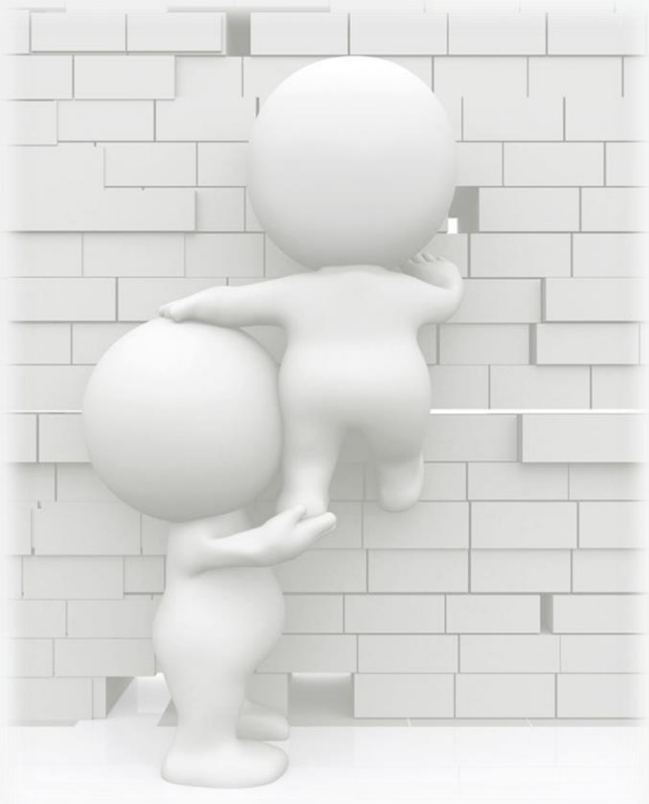
Not talking about these today ☺

Network & Host IDS/IPS

- NIDS – Network Intrusion Detection System
  - Using a network tap, span port, or hub collects packets on the network
  - Attempts to identify unauthorized, illicit, and anomalous behavior based on network traffic
  - Methods: signature file comparisons, anomaly detection, stateful protocol analysis
  - Using the captured data, the IDS system processes and flags any suspicious traffic
- HIDS – Host Intrusion Detection System
  - Generally involves an agent installed on each system, monitoring and alerting on local OS and application activity
  - Attempts to identify unauthorized, illicit, and anomalous behavior on a specific device/OS
  - The installed agent uses a combination of signatures, rules, and heuristics to identify unauthorized activity
- IDS vs IPS?
  - An IDS only has passive role (identify, log and alert)
  - An IPS also blocks network traffic

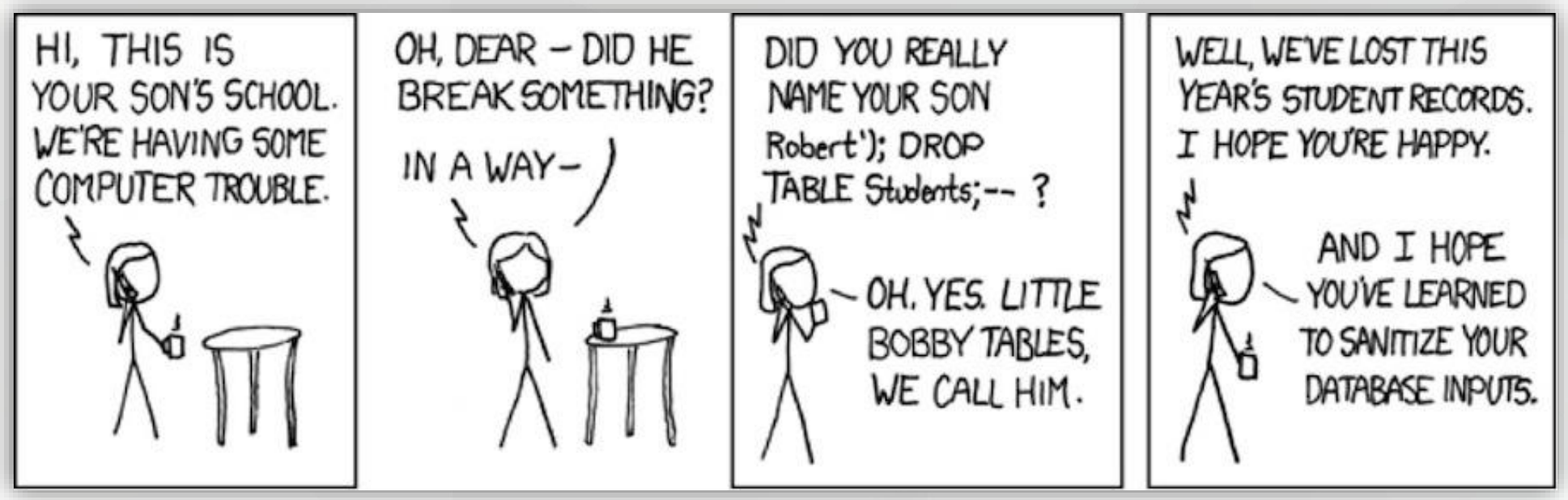
NIDS Design Considerations

- Logical Target of Attacks
  - Each component a potential point of vulnerability and hence attacks
- Possible Attacks on their
  - “Availability” (total shutdown)
  - “Accuracy” (false positives)
  - “Completeness” (false negatives)
- Need to be Reliable, Robust
  - Avoid false sense of security
- There’s no IDS out there with 100% accuracy
  - They all generate both false positives and false negatives



Problems with NIDS

- Passive Network Monitors
  - Inherently “fail-open”
  - Cease to provide protection when subverted
- Vulnerability to Denial of Service
  - Process all flows to all protected end-systems
  - Being complex systems require lots of resources
  - Resource starvation problem is not easily solvable
- Insufficient Information on the Wire
  - Not enough to correctly reconstruct the state of complex protocol transactions like it is done at the end-systems
- Diversity in Protocol Implementations
  - Packet processing differs across end-systems
  - Leads to ambiguous interpretations
- Unknown Internal Network Conditions
  - Topology, router configs, traffic congestion, etc.



Advanced Evasion Techniques

Advanced Evasion Techniques (AETs)

- Delivered in a highly liberal way
- Security devices are designed in a conservative way
- Employ rarely used protocol properties
- Use unusual combinations
- Create network traffic that disregards strict protocol specifications
- Exploit the technical and inspection limitations of security devices: memory capacity, performance optimization, design flaws

Some of the most common attacks against NIDS

- Insertion
  - Stuffing the analyzer with “invalid” packets
- Evasion
  - Slipping “valid” packets past the analyzer
- Denial of Service (DoS)
  - Causing resource starvation and overloading the IDS
- Pattern-matching weaknesses
  - Exploiting pattern-based detection approach employed by most IDS
  - Pattern-matching weaknesses
- Encryption and tunneling
  - SSL, SSH, IPSec, etc.
- Fragmentation
  - Breaking the attack into multiple packets
- Protocol violation
  - Attacks targeted at complex protocol (e.g. SMB)
- File Locations and Integrity
  - Circumventing triggers in HIDS
- Application Hijacking
  - If done correctly few HIDS will detect it, while NIDS usually skip the application layer completely

Insertion

Occurs when the NIDS is **less strict** in processing packets than the internal network

NIDS accepts packets that the end system rejects or doesn't even receive

Attacker's data stream

2	3	3	5	4	1	6
T	T	X	C	A	A	K

NIDS data stream

Accepts 3rd packet which overwrites 2nd packet data
Interprets **ATXACK**

1	2	3	3	4	5	6
A	T	T	X	A	C	K

End system data stream

Rejects 3rd packet for some reason (or doesn't even receive it, i.e. TTL)
Interprets **ATTACK**

1	2	3	3	4	5	6
A	T	T	X	A	C	K

Evasion

Occurs when the NIDS is **more strict** in processing packets than the internal network

An end-system can accept packets that the NIDS rejects

Attacker's data stream

2	3	3	5	4	1	6
T	X	T	C	A	A	K

NIDS data stream

Rejects 3rd packet for some reason and sees **ATXACK**

1	2	3	3	4	5	6
A	T	X	T	A	C	K

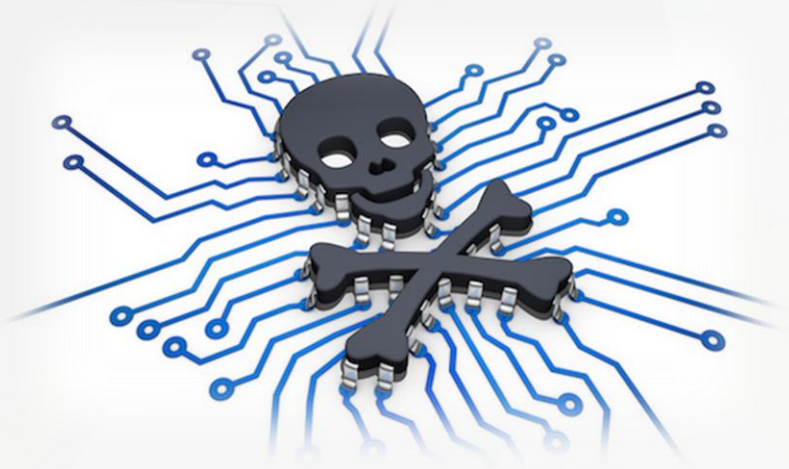
End system data stream

Accepts 3rd packet which overwrites 2nd packet
Interprets **ATTACK**

1	2	3	3	4	5	6
A	T	X	T	A	C	K

Real insertion / evasion attacks

- Mostly exploit basic network and protocol ambiguities at the NIDS
  - Ambiguous interpretation of header fields
  - Ambiguous handling of header options
  - Ambiguous fragment/segment reassembly
- NIDS and the end-system get different views of the same data stream
  - Ambiguities cause NIDS to accept/reject packets differently than the end-systems
- Differences in protocol implementations
  - Non-conformance to Protocol Standards
  - Every OS has a different protocol stack
- End-system and router configurations
- Application/Socket level options



NIDS ambiguities - examples

Related field	Ambiguity (problem) for the NIDS
TTL	Will the packet reach the end-system before TTL becomes 0?
Length, DF	Will all downstream links be able to transmit this big packet without fragmenting it (Don't Fragment -DF- bit set)?
IP Option(s)	Will the end-system/routers accept packet with specific IP option(s)? E.g. (Strict) Source Route option
TCP Option(s)	Will the end-system accept packets with specific TCP option(s)?
Data	Will the end-system accept data in a SYN packet?
ToS	Does the packet conform to all internal routers (DiffServ)?
IP Frag Offset	How will the end-system reassemble overlapping fragments?
TCP Seq No	How will the end-system reassemble overlapping segments?

Session Splicing

- Technique used to bypass IDS where an attacker splits the attack traffic into many packets such that no single packet triggers the IDS
  - It is effective against IDS-es that do not reconstruct packets before checking them again through intrusion signatures
- If attackers are aware of delays in packet reassembly at the IDS, they can add delays between packet transmissions to bypass the reassembly
  - Many IDS-es stop reassembly if they do not receive packets within a certain time
  - IDS-es become useless if the target host keeps sessions active for a longer time than the IDS reassembly time
- Any attack attempt after a successful splicing attack will not be logged by the IDS

Fragmentation

- Similar to Session Splicing
  - Attackers send packets in blocks that do not trigger IDS signatures or cause alerts
  - Generally more powerful than Session Splicing
- Two common fragmentation methods
  - One method overwrites a section of a previous fragment
  - The second method overwrites a complete fragment
- Enables attackers to write an entire packet of garbage information and craft their attack to blend in with standard protocols
- Some IDSs do have ways to handle these attacks through reassembly

Fragmentation - examples

Attack 1: Overlap Method

- Packet 1: **GET /cgi-bin/**
- Packet 2: **aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa/../phxx**
- Packet 3: **f?**

- This example of fragmentation can overwrite the 'xx' portion of Packet 2 with the data in Packet 3, making the information resemble the following:
  - GET /cgi-bin/ aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa/../phf?

Attack 2: Overwrite Method

- Packet 1: **GET /cgi-bin/**
- Packet 2: **some_normal_filename.cgi**
- Packet 3: **/aaa/../aaa/../aaa/../phxx**
- Packet 4: **f?**

- This example is similar to the first, but in this example, the 'xx' portion is overwritten and the some_normal_filename.cgi packet is completely overwritten with the last two packets
  - This leaves **GET /cgi-bin/phf?** as the end result.

Denial of Service (DoS) & False Positive Generation

- Basic problem
  - NIDS needs to simulate the operation of all protected end-systems and internal network
  - Scarce Resources (CPU cycles, memory, disk space, bandwidth)
  - Usually working in a “fail-open” state
- CPU DoS (target computationally expensive operations)
  - Fragment/Segment reassembly
  - Encryption/Decryption
- Memory DoS (target state management operations)
  - TCP 3-way Handshake (TCP Control Block - TCB)
  - Fragment/Segment reassembly
- Network Bandwidth DoS (target NIDS’s inability to process packets at line speed)
- Reactive Systems DoS
  - Trigger lots of alarms (false positives)
  - Prevent valid access by spoofed addresses
  - Hide real attacks

Pattern-Matching Weaknesses

- Most IDS solutions employ a pattern-based detection component
- This approach is problematic, because not all input needs to be the same to trigger vulnerabilities

- Example pattern:
  - **GET /cgi-bin/phf?**
- Obfuscation:
  - **GET /cgi-bin/aaaaaaaaaaaaaaaaaaaaaaaaa/..%25%2fp%68f?**

- Both versions result in the same output, yet look very different

- Unicode evasion can also be used here
  - e.g. **** can be represented as **5C**, **C19C** and **E0819C**

Protocol Violation

- Attacks can be targeted at complex protocols
  - e.g. SMB (Server Message Block), MSRPC, SunRPC
- In order to provide protection to a complex protocol, the IDS has to have a deep understanding of it
- The IDS implementation also needs to be
  - Fault-tolerant
  - Resilient
  - Able to cope with excessive and unexpected connections and requests

TTL Attacks

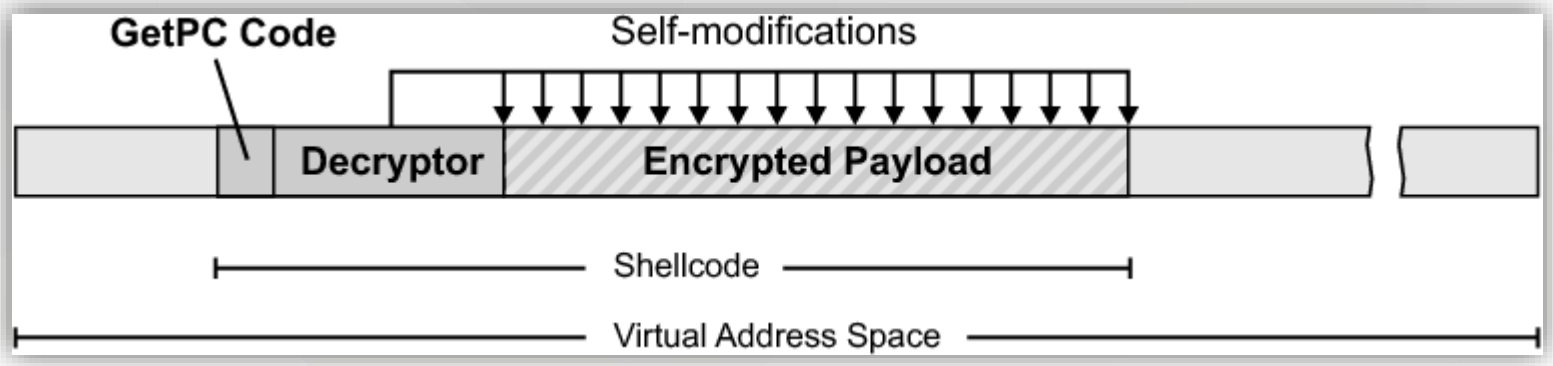
- For TTL attacks, attackers must have some knowledge of the internal network topology
  - Attackers must know the distance to the end host and whether an IDS is placed in front of the end host
- By using a small TTL flag in a TCP packet, attackers can send packets that will only reach the IDS and not the end host
  - The IDS, in turn, will think the packet addressed to the end host will make it there
  - This allows attackers to inject garbage packets into the IDS stream processing
- Example:
  - Packet 1: **GET /cgi-bin/p** TTL 15
  - Packet 2: **some_file.cgi?=** TTL 10
  - Packet 3: **hf?** TTL 15
- This assumes that the end host is beyond the 15 TTL limit and will receive the data
- It also assumes that the IDS is within the 10-14 TTL limit, and any data lower than that will not reach the destination host
  - The IDS receives **"GET /cgi-bin/psome_file.cgi?=hf?"**
  - The end host will receive **"GET /cgi-bin/phf?"**

Urgency Flag

- The urgency flag is used within the TCP protocol to mark data as urgent
  - When the urgency flag is set, all data before the urgency pointer is ignored, and the data to which the urgency pointer points is processed
- Some IDSs do not take into account the TCP protocol's urgency feature
  - Attackers can place garbage data before the urgency pointer
  - The IDS reads that data without consideration for the end host's urgency flag handling
  - This means the IDS has more data than the end host actually processed
- Example of an urgency flag attack:
  - "1 Byte data, next to Urgent data, will be lost, when Urgent data and normal data are combined."
  - Packet 1: **ABC**
  - Packet 2: **DEF** Urgency Pointer: 3
  - Packet 3: **GHI**
  - End result: **ABCDEFHI**
- According to the 1122 RFC, the urgency pointer causes one byte of data next to the urgent data to be lost when urgent data is combined with normal data

Polymorphic Shellcode

- Most IDSs contain signatures for commonly used strings within shellcode
  - This is easily bypassed by using encoded shellcode containing a stub that decodes the shellcode that follows
  - This means that shellcode can be completely different each time it is sent
- Polymorphic shellcode allows attackers to hide their shellcode by encrypting it in a simplistic form
  - It is difficult for IDSs to identify this data as shellcode
- This method also hides the commonly used strings within shellcode, making shellcode signatures useless



ASCII Shellcode

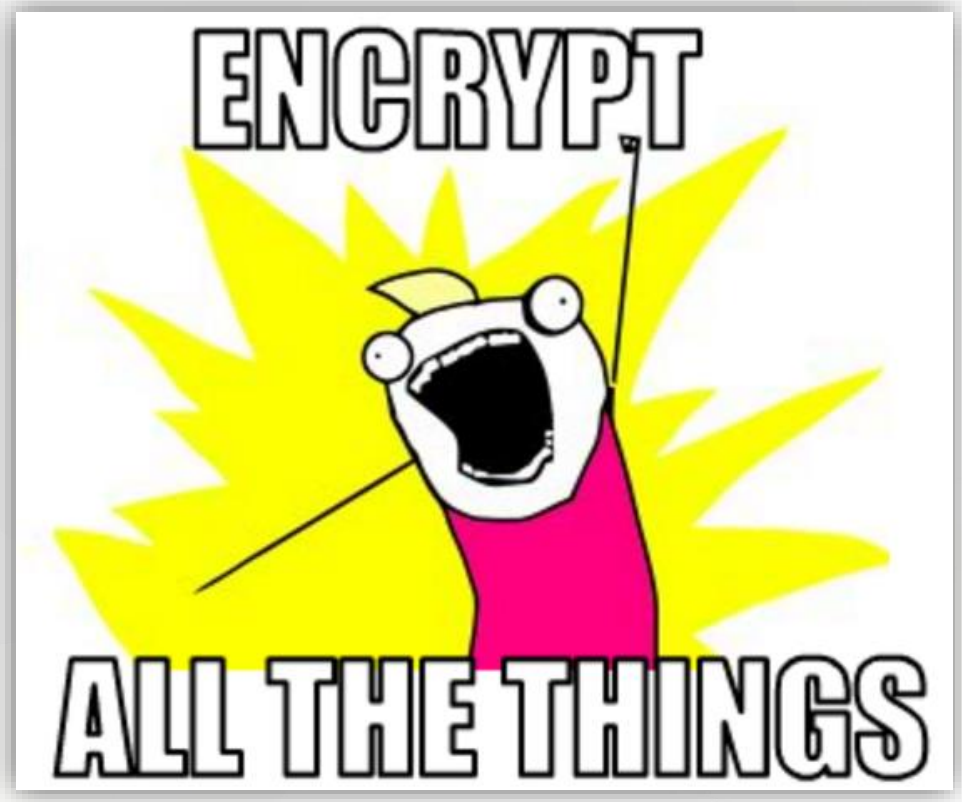
- Similar to polymorphic shellcode
  - ASCII shellcode contains only characters contained within the ASCII standard
- This helps attackers bypass IDS pattern matching signatures as strings are hidden within the shellcode in a similar fashion to polymorphic shellcode
- The following is an ASCII shellcode example:

```
char shellcode[] =
"LLLLYhb0pLX5b0pLHSSPPWQPPaPWSUTBRDJfh5tDS"
"RajYX0Dka0TkafhN9fYf1Lkb0TkdjfY0Lkf0Tkgfh"
"6rfYf1Lki0tkkh95h8Y1LkmjpY0Lkq0tkrh2wnuX1"
"Dks0tkwjfX0Dkx0tkx0tkyCjnY0LkzC0TkzCCjtX0"
"DkzC0tkzCj3X0Dkz0TkzC0tkzChjG3IY1LkzCCCC0"
"tkzChpfcMX1DkzCCCC0tkzCh4pCnY1Lkz1TkzCCCC"
"fhJGfXf1Dkzf1tkzCCjHX0DkzCCCCjvY0LkzCCCjd"
"X0DkzC0TkzCjWX0Dkz0TkzCjdX0DkzCjXY0Lkz0tk"
"zMdgvvn9F1r8F55h8pG9wnuvjrNfrVx2LGkG3IDpf"
"cM2KgmnJGgbinYshdvD9d";
```

- When executed, the shellcode above executes a "/bin/sh" shell

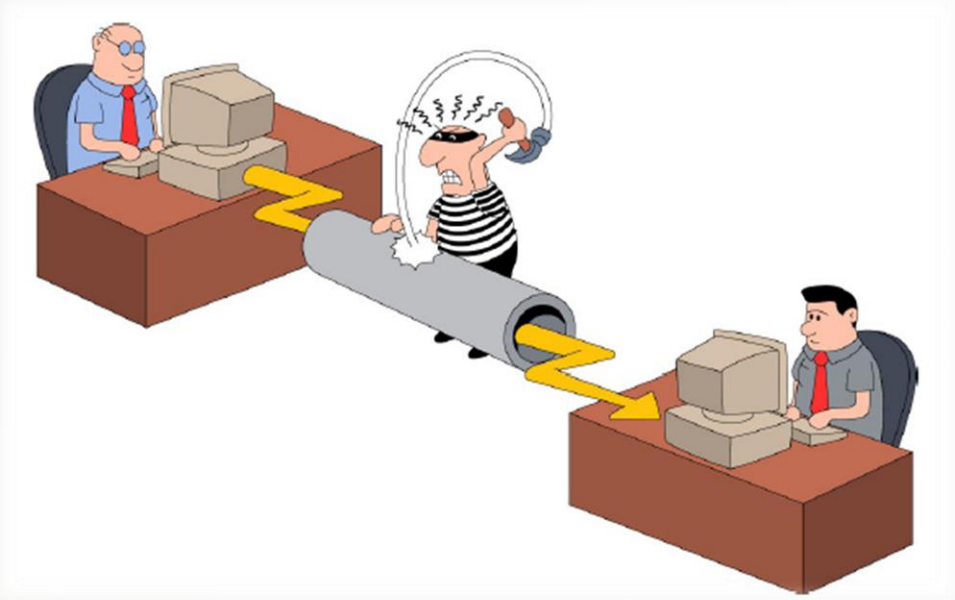
Encryption and Tunneling

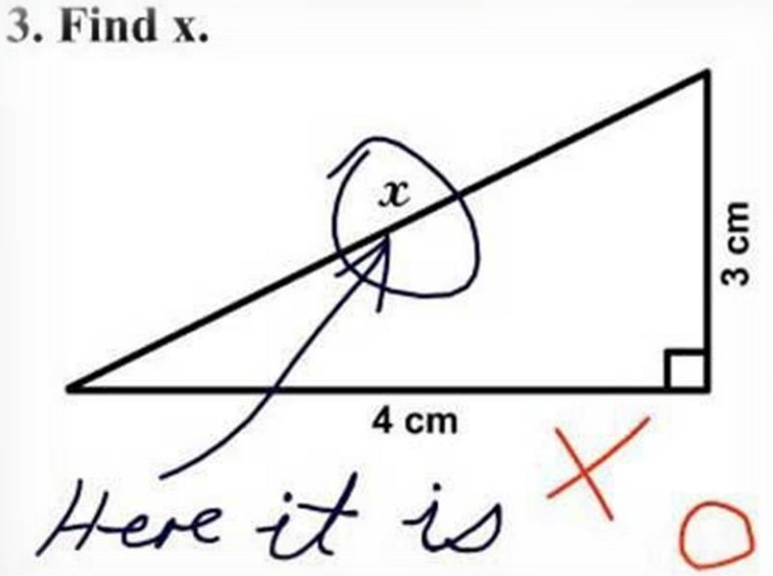
- When the attacker manages to establish an encrypted tunnel to the target, IDS-es are evaded completely
  - Any sort of encrypted connection/tunneling works
  - SSH
  - SSL
  - IPSec
  - RDP
    - etc.



Application Hijacking

- Application layer attacks enable many different forms of evasion
- Many applications that deal with media such as images, video and audio employ some form of compression
- When a flaw is found in these applications, the entire attack can occur within compressed data, and the IDS will have no way to check the compressed file format for signatures





Potential Solutions

Normalization

- Normalization takes obfuscated input and attempts to translate it into what the end host will eventually see
  - This usually entails encoding in formats such as Unicode and UTF8
- The normalization process allows for encoding, translation and the application of pattern matching to the normalized data
  - Prevents attackers from obfuscating the attack strings using Unicode or UTF8 strings
  - Polymorphic shellcode & ASCII shellcode could circumvent this
- Some IDSs are attempting to apply normalization to polymorphic shellcode
  - CPU intensive, which affects monitoring for the remaining network traffic
- Normalization also applies to network data
  - Some IDSs normalize fragmented packets and reassemble them in the proper order
  - This enables the IDS to look at the information just as the end host will see it
- In addition, some IDSs change the TTL field to a large number
  - This ensures that packets reach the end host

Packet Interpretation Based on Target Host

- IDS-es are at a disadvantage
  - These systems attempt to recreate what the end host will see and handle
  - There are a lot of disparate methods of communicating data over a network
- The end host's TCP/IP stack should be used (host-based IDS)
  - Better than trying to recreate the stream in a way that the stream may be handled
  - In using the host to do the work, the guessing portion of the task is eliminated
- Another option: using modular TCP/IP stacks within an IDS
  - Using the stacks based on the targeted host's operating system.
  - Specific OS handling of anomalous traffic must be thoroughly reviewed
  - Effective in mitigating fragmentation, RST packet handling and Urgency Flags

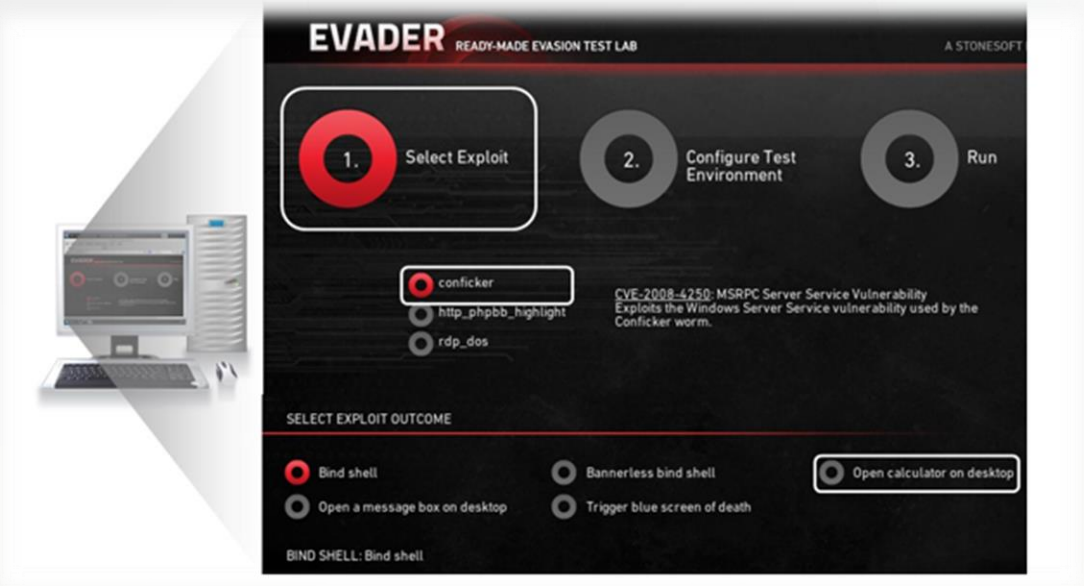
Tools & Resources

Some free IDS out there…

- ACARM-ng
- AIDE (Advanced Intrusion Detection Environment)
- Bro NIDS
- Fail2ban
- OSSEC (Open Source Host-based IDS)
- Prelude SIEM (Security Information & Event Management)
- Samhain
- Snort
- Suricata
- Tripwire

...and some IDS evasion tools

- Evader
- Nmap
  - Nmap reference on IDS evasion
- libemu
- Kali Linux
  - Fragroute
  - Fragrouter
  - InTrace
  - SniffJoke
- Other tools
  - Wireshark
  - HxD (hex editor/viewer)



Closing remarks

Upcoming Public Courses

Avælgo Training

IT Security Awareness Seminar & Web Application Security Seminar

Bucharest, 10-11 December 2015

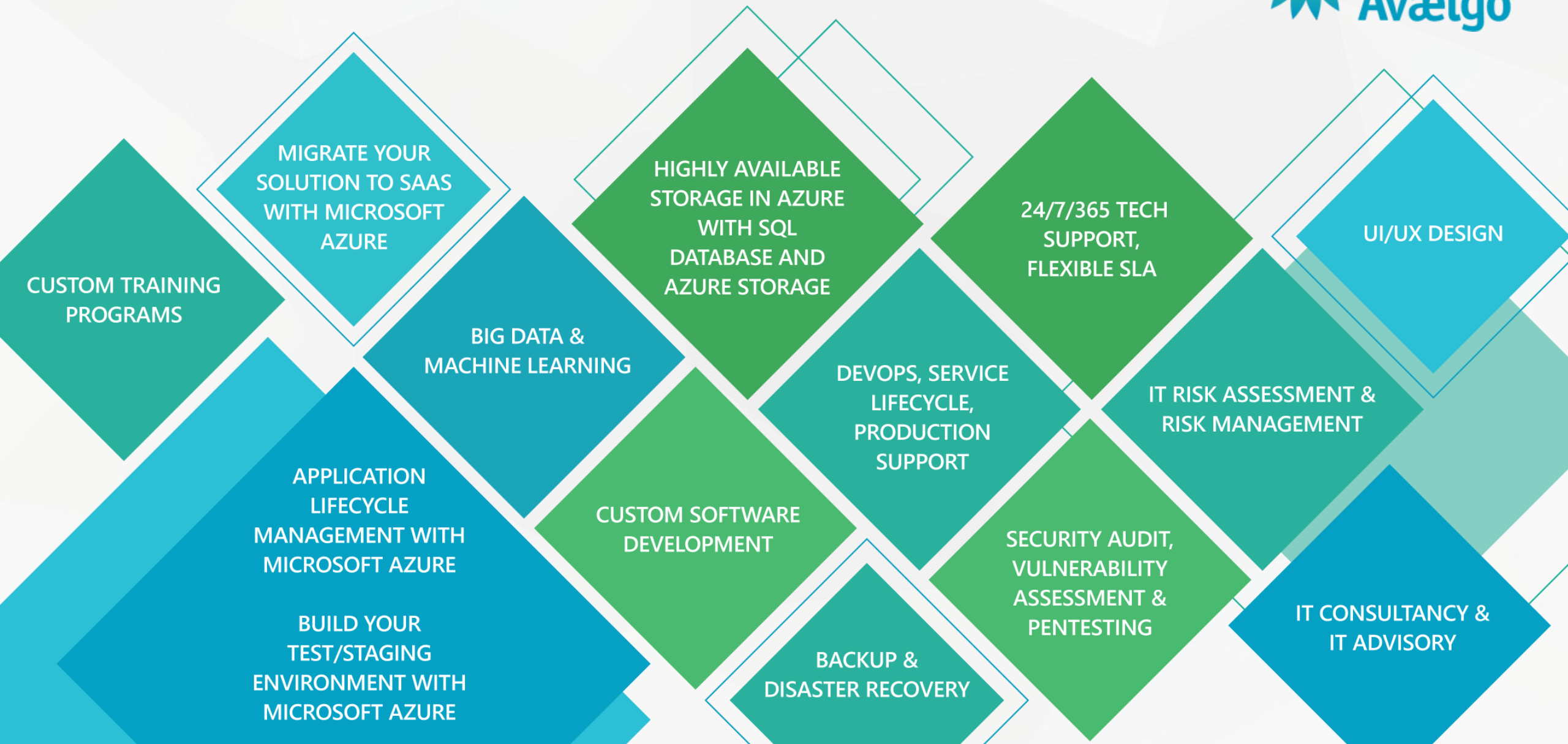
Troubleshooting Windows Infrastructure - From Zero to Hero Masterclass

Bucharest, 11-15 January 2016 CQURE ACADEMY

Paula Januszkiewicz is an IT Security Auditor and Penetration Tester, Enterprise Security MVP and trainer (MCT) and Microsoft Security Trusted Advisor. She is also a top-speaker on many well-known conferences (for example: TechEd North America, TechEd Europe, TechEd Middle East, RSA, TechDays, CyberCrime etc.) and writes articles on Windows Security. She proudly holds the role of the Security Architect at IDesign and she drives her own company CQURE. She conducted hundreds of IT security audits and penetration tests, including governmental organizations. Her distinct specialization is definitely on Microsoft security solutions in which she holds multiple Microsoft certifications (MCITP, MCTS, MCSE, MCDBA etc.) besides being familiar and possessing certifications with other related technologies. Paula is passionate about sharing her knowledge with others. In private, she enjoys researching new technologies, which she converts to authored trainings. She has access to a source code of Windows.

Your Technology Business Partner

Avælgo



Tudor Damian

General Manager & IT Solutions Specialist @ Transcent

Operations Manager & Partner @ Avaelgo

Co-founder @ ITCamp & ITCamp Community

- Certified Ethical Hacker (EC-Council)
- Certified Security Professional (CQURE)
- Cloud & Datacenter Management MVP (Microsoft)

Contact: tudy@tudy.ro / www.tudy.tel

ITCAMP TRANSCENT YOUR IT DEPARTMENT Avælgo C|EH™ Certified Ethical Hacker MVP Microsoft® Most Valuable Professional