2006

# Towards Attack-Agnostic Defenses

David Brumley
*Carnegie Mellon University*

Dawn Song
*Carnegie Mellon University*

# Towards Attack-Agnostic Defenses

David Brumley and Dawn Song
*Carnegie Mellon University*
{dbrumley,dawnsong}@cs.cmu.edu

## Abstract

Internet attackers control hundreds of thousands to perhaps millions of computers, which they can use for a variety of different attacks. Common attacks include spam delivery, phishing, and DDoS. The current research community focus is on defenses for each specific attack type compromised hosts may launch. However, attack-specific approaches almost always have two fundamental drawbacks: they do not address the root problem that attackers control an army of compromised hosts, and they do not provide the right incentives for users to properly secure their machines. As a result, attack-specific defenses may be defeated by new attacks, even those that may be only slightly different from old attacks.

We argue researchers should also focus on attack-agnostic defenses whose effectiveness does not depend on the particular attack type. We initiate this line of research by investigating the design space for attack-agnostic defenses, and then detailing two extreme points within the design space: an Internet Watch List and an Internet Reputation System.

## 1 Motivation

Today, attackers control thousands to perhaps millions of compromised hosts [9], and are constantly attempting to compromise more. The compromised hosts can be used to launch a variety of attacks, such as spam, phishing, spyware, worms, DDoS, etc. Sophisticated attackers use compromised hosts essentially as massive distributed computing or content distribution platforms. The availability and number of uses for these large distributed platforms of compromised hosts has created an entire black-market industry for renting, trading, and managing "owned" hosts, leading to economic incentives for attackers to compromise additional hosts.

How can we address the problem of such a vast army of hosts controlled by attackers? Current research often focuses on defenses for each individual form of attack, such as "Anti-spam", "Anti-phishing", "Anti-spyware", and "Worm defense" systems. Attack-specific solutions and research are certainly valuable, but usually have two fundamental drawbacks:

- **Do not address the root problem.** The root problem is that attackers control hosts, sometimes in networks up to 100K+, which then manifests into many different kinds of attacks. Attack-specific solutions tend to depend on attack details that make the solution ineffective when the compromised hosts are used for new or different nefarious purposes. For example, a computer blacklisted for relaying spam can still be used to gather financial information in phishing attacks.
- **Do not give the right incentives.** Users have very little motivation to patch vulnerable services or fix a compromised host. Users are reluctant for good reason: if a compromise does not affect them, then why should they spend the time and effort fix the problem? Even technologically sophisticated users suffer from this problem, since fixing a compromised host is time consuming with few if any direct benefits. On the other hand, attackers have a huge (and often economic) incentive to compromise hosts. This imbalance tips security in the attackers favor.

**Research Challenge: Attack-Agnostic Defenses.** We pose a fundamental challenge to the research community: can we develop Internet-wide *attack-agnostic* defense systems? Attack-agnostic defenses should offer holistic protection. Attack-agnostic defenses do not rely on specific attack details to fend off attacks from known malicious nodes (though they may use attack-specific detection mechanisms, but only to initially identify malicious nodes).

Attack-agnostic defenses not only address the root problem, but also provide incentives for users. The usability of a host is tied to how well it is maintained and whether it is compromised. Thus, users have direct motivation to patch vulnerable services and quickly fix compromised nodes.

To initiate research, we investigate the design space and several important properties for attack-agnostic defense systems. This research is motivated by a previous effort for building the next generation secure internet [5] in which Tsudik proposed the concept of conditional privacy for nodes on the network. We consider two extreme design points: an Internet Watch List (IWL) and an Internet Reputation System as two concrete examples. They illustrate many of the trade-offs and challenges in the attack-agnostic defense system design space.

## 2 Design Space of Attack-Agnostic Defenses

An attack-agnostic defense system should defend against malicious nodes regardless of the type of attack. The abundant research for detecting specific attacks can be used to identify misbehaving nodes, but then what? How how can we leverage this information to defend against subsequent and possibly different attacks? Candidate approaches include:

**Access Control Lists.** Access control lists (ACL) can be used to grant or restrict traffic on the network, e.g., filter traffic from known compromised nodes. Currently, ACL's are widely used for attack-specific defenses, such as spam black-holes, filtering vulnerable services, etc. One way to use ACL's in an attack-agnostic defense is to filter all traffic from a node participating in any malicious activity, therefore the attacker cannot simply switch attack vectors.

**Capabilities.** Capabilities, like ACL's, can be used to grant or restrict communication rights on the network. Capabilities differ from ACL's because recipients need only check that the capability is valid, there is usually no need for additional access validation steps based on the senders identity. For example, SIFF [14] uses network-based capabilities to prioritize packets subject to the recipients control to defend against DDoS attacks. One attack-agnostic approach leveraging capabilities is to require all senders to acquire an appropriate capability before sending a message to a particular recipient.

**Reputation.** The reputation of a node may be used to establish how likely it is to send malicious traffic. For example, if a node participates in any type of malicious activity, its reputation is diminished. An attack-agnostic defense system may institute policies based upon the reputation, e.g., filter traffic or reduce bandwidth from nodes with low reputation. Although reputation systems suffer a somewhat lackluster reputation themselves, they have recently gained traction in several areas, e.g., self-moderated websites such as Craigslist and Slashdot, securing BGP [15], and establishing trust in P2P systems [6, 8, 13].

**Constraints.** Constraints may be used to determine what properties of a node are likely important when deciding whether a node is malicious or not. For example, a constraint may specify that a node is willing to communicate with any other node that has been patched within the last day. Constraints differ from reputations because constraints are specified by a node, while reputations are essentially given to a node. For example, Off-by-default proposes a system where end-hosts signal and routers exchange reachability constraints [4].

## 3 Common Properties of Attack-Agnostic Defense Systems

Many different attack-agnostic systems share similar properties and design points.

**Identity.** Most attack-agnostic defense approaches require (or are greatly simplified by) a way to identify misbehaving nodes. ACL's filter based upon identity. Capabilities may be granted in part based upon the remote node's identity, e.g., whether the remote node is known to be malicious. Reputations are essentially an implicit identity because although they are non-unique, they can be used to persistently identify particular classes of nodes. Constraint-based defenses similarly can be used to identify particular classes of nodes, e.g., those that satisfy particular constraints.

In current defense systems, a node's identity is often assumed to be its Internet address, usually for backwards compatibility reasons. However, overloading an address as an identity causes many problems. One problem is two hosts may share the same address, thus the same identity. For example, malicious and benign hosts behind the same NAT cannot be easily distinguished based on address alone. Another problem is that a single host may have several addresses thus several identities. For example, a malicious node may be able to switch to a new address, perhaps with a different ISP, each time its current address is identified as malicious.

**Privacy.** The need to identify nodes, whether it be only misbehaving nodes or all nodes, should be balanced against privacy considerations. For example, simply providing a unique identifier to each node on the Internet allows malicious nodes to be persistently identified, but may also allows malicious parties to profile nodes. How can we balance privacy considerations among nodes with the ability to uniquely identify them? Should malicious nodes have less privacy than benign nodes? Once a node is fixed, how can privacy be restored?

**Policy Diversity.** Attack-agnostic defenses protect against malicious nodes, but different sites may disagree on what constitutes malicious action. On the one hand, we would like to leave the decision to each node. On the other hand, we would like nodes to be able to share information about malicious nodes. For example, one site may wish to inform another site about known DDoS zombies. What if the other site has no policy regarding DDoS? What if the other site has different criteria for determining when a node is a zombie? Thus, whether and how the system deals with policy diversity among the participants is an important design consideration.

**Security.** The defense system should work correctly, even against an adversary attempting to exploit the defense system itself. As defense systems become more powerful and more autonomous, they become attractive targets. For example, reputation-based systems should protect against framing attacks where one node fraudulently decreases another node's reputation. In essence, the defense system should answer the question "why and in what scenarios is this secure?"

The security of a system depends not only on communication security, but also the auditability and accountability of the authorities. More secure defense systems will allow for privileges to be audited for potential misuse.

**Practicality.** There are several systems considerations to make defense systems practical. Defense systems should have high performance, be scalable, and be robust against attacks. In addition, the defense system should work (at least for adopters) even when partially deployed.

# 4 Internet Watch Lists

In this section, as an example, we investigate one extreme design point for attack-agnostic protection: privacy-preserving Internet Watch Lists (IWL). The IWL approach aims to provide an Internet-wide primitive which allows one node to identify known malicious nodes, but does not leak any information about nodes not on the list. The IWL highlights many of the trade-offs described in the previous section. In particular, we construct a privacy-preserving IWL that trades off privacy, identity, and security considerations.

## 4.1 IWL Actions

An IWL supports 3 fundamental actions: *node addition*, *node removal*, and *node identification*. Addition is performed when a node misbehaves. Node removal removes a node from the IWL when it is no longer malicious, e.g., it has been patched, fixed, or upgraded.

Node identification allows a node to identify other known malicious nodes, e.g., a particular packet is sent by a node known to be compromised. The recipient can then take appropriate action. One action may be to block anyone on the IWL, while another may be to enable additional protection measures such as supervised execution. The IWL should not reveal any information about senders not on the list.

At first glance, implementing these three actions may seem straightforward. One straw-man solution is to assign each host a unique identifier (or leverage one of the many unique identifiers already on the host such as the OS license key, the CPU serial number, etc.) and require the identifier appear on all packets sent. The IWL is a list of known malicious ID's, which recipients match against packets. However, the straw-man solution does not provide users any privacy.

## 4.2 Background: Group Signatures

We use a special cryptographic construction called group signatures [3, 7] to provide conditional privacy to nodes. A group signature scheme allows each member of a group of players to sign messages on behalf of the group. Given these signatures, no player or coalition of players (except the trusted *group manager*) can distinguish the player that produced any signature, nor can they determine if two signatures were produced by the same group member. A group signature scheme has four procedures: `Initialize, Sign, Verify, Open`. The properties of a group signature scheme are as follows:

- *Correctness.* Signatures produced by any group member (using the `Sign` algorithm) must be accepted as valid signatures by the verification algorithm, `Verify`.
- *Unforgeability.* No adversary who is not a group member can produce signatures on behalf of the group that are accepted by `Verify`.
- *Anonymity.* For any valid signature of any message, it is computationally difficult to determine which group member produced it.
- *Exculpability.* No player, including the group manager, can sign on behalf of any other player.
- *Traceability.* The group manager can always identify the party that produced any valid signature, even if a coalition of dishonest players works together to produce the signature (*coalition-resistance*).

## 4.3 A Privacy-Preserving IWL

We show how to create a privacy-preserving IWL. Each node is given an identity via an *Identity Authority* (IA) in the form of a group signing key. The IA itself acts as the group manager. We outline the important concepts below (a more formal treatment is beyond the scope of this paper).

**Identity Initialization.** The IA creates a group public key $gpk$ which is known to everyone and the corresponding group private key which is only known to the group manager (the IA). For each computer $n$, the IA runs the `Initialize` algorithm with the computer, which outputs a secret group signing key $gsk_n$, and a secret identification key $gfk_n$ for the node. The corresponding secret signing key $gsk_n$ is installed into the node's TPM. The IA retains the corresponding identification key $gfk_n$.

One property of group signatures is the IA cannot sign on behalf of a node even though it has corresponding identification key. Additionally, a node's secret group signing key is constructed based upon an additional secret known only to the node, such that the IA never has possession of $gsk_n$.

Note that in our scheme node removal is accomplished by re-running the `Initialize` algorithm on the fixed host, yielding a new key, i.e., a new identity.

**Message Signing.** When a node sends a message, it performs the group signature `Sign` operation with its signing key $gsk_n$ and identification key $gfk_n$. Group signatures have the privacy preserving property that anyone can verify that the resulting signature is valid, but does not reveal *which* node created the signature. For now, we assume every packet is signed.

**Verification.** When a recipient receives a signed packet, it runs the group signature `Verify` algorithm to prove the sender is a member of the group. Note that non-group members are able to verify as well since the verification key is public.

**Identification.** Suppose a receiver receives a malicious message $m$. A valid signature indicates the message was sent by a member of the group, but the receiver does not know which one. Here, the receiver submits $m$ to the IA, which checks that $m$ is malicious and runs the `Open` algorithm. `Open` returns the corresponding identification key $gfk_i$ of the sender. The IA then publishes $m$ and $gfk_i$ publicly. The public identification key allows any recipient to check if subsequent packets were signed by the malicious node. Nodes with published identification keys are considered part of the IWL since any site can exclude them from participating in future transactions.

## 4.4 Challenges and Directions

A privacy-preserving IWL is one extreme attack-agnostic design point. It attempts to balance security, privacy, and the identity properties raised in Section 3. In this section, we discuss some of the remaining challenges.

**Challenge: Incremental Deployment.** Our scheme can be deployed today using existing technology. Although IPV4 doesn't provide a signature field, we can incorporate the signature as part of the transport layer. Initially, hosts that participate in the scheme can be given a higher QoS to motivate adoption. This makes sense: hosts not on the IWL are verified as uncompromised, thus are less likely to initiate remote attacks. An open question is what other mechanisms are available to help speed along incremental deployment.

**Challenge: IA Scalability, Performance and Reliability.** The IA is a potential bottleneck for the IWL. However, standard techniques from cryptography and fault-tolerant research can be used to a) protect the group master secrets and b) replicate the IA to ensure performance and reliability. Adopting these techniques to our problem domain may yield additional benefits. For example, the IA itself can be divided into three separate authorities which can be protected and replicated independently: an issuing authority, a revocation authority, and the IWL itself. Each of these components may have particular requirements that make it amenable to further optimizations. How should we apply these techniques to create a cohesive, secure, and reliable architecture?

**Challenge: IWL Accuracy, Scalability, and Performance.** We note that one advantage of our approach is hosts are never taken off the watch list, thus watch list data is never stale. Over the long run, however, we would likely want to prune the list for efficiency reasons. This can be done in batch by IA's: any node that has been reassigned a group key can have the old one removed from the list.

We must also address how quickly the watch list can be updated. We believe existing technology, e.g., CDN's, P2P networks, etc., provide a starting point to a solution, but do not solve all problems. For example, should the entire list be replicated everywhere, or should there be a central "master" authority?

**Challenge: Informing End Nodes.** We should proactively inform end-nodes when they are on the IWL. For example, bank websites can exclude nodes on the IWL from managing accounts. The bank can inform the user at the same time that they are on the watch list. Similarly, services such as Google which may not care whether a node is on the watch list or not can certainly still inform users when they visit the site.

**Challenge: End-Node Performance.** We use group signatures as a fundamental primitive. Although we describe the IWL where each packet is signed with a group signature, it is possible to augment the scheme so that a signature is used only to authenticate the first packet, and a more efficient primitive is used for subsequent packets.

We note that existing network architectures already incorporate cryptographic operations for every packet, e.g., WEP, HIP [10], and mobile IPv6 [11], thus requiring end-nodes to sign packets does not appear to make our approach impractical. However, it would be interesting to find a lighter-weight primitive that confers the same benefits as group signatures.

**Challenge: Federated Authorities.** The current Internet consists of a collection of federated authorities, with a few centralized authorities that make management-only decisions. For example, addresses and top-level domains are both managed by a centralized top-level organization, but day-to-day management is left to local federations. We can create a similar hierarchy using techniques borrowed from PKI, e.g., larger groups are upper layers in the tree, which delegate authority to assign sub-groups to lower layers.

**Challenge: Adding Nodes to the IWL.** How do we decide when to add a node to the IWL? Most policies are straightforward: packets containing viruses, malware, and worms can all be distinguished with simple rules. Other policies, such as what constitutes spam, may not have universal rules. How should we deal with such criteria? If federated authorities are used, how do we deal with disagreement among constituents?

The policies for implementing these rules is also an interesting research topic. There are several possibilities:

- **Rule-based.** The IA publishes a set of rules for releasing the filtering key. When a recipient claims a packet is malicious, he proves to the revocation authority those standards are met by showing the packet matches the corresponding rule. Although technologically unsophisticated, this approach seems promising in cases where immediate action is important. For example, during a worm outbreak the revocation authority could post a worm signature. Packets matching the worm signature are considered malicious so that the filtering key is exposed automatically.
- **Time-based.** The group signature scheme is augmented so that the keys evolve over time. This is implemented by giving each host a set of signing keys $s_1, s_2, .., s_t$ such that at time interval $i$ only signing key $s_i$ is valid. Similarly, the revocation authority maintains a set $f_1, f_2, ..., f_n$ of filtering keys for each time interval [12].
  Anyone can request the current filtering key for a host, which they can use to immediately filter all traffic. However, the filtering key will eventually expire. This approach can be paired with rule-based filtering. Here, time-based filtering is used as a stop-gap when needed until the appropriate IA can be convinced to release the permanent filter key.
- **Threshold-based.** In this scheme, the filtering key is split onto $n$ revocation servers such that $k$ servers need to be convinced a packet is malicious in order to run `Open` to get the filtering key for a node. Such schemes can be implemented directly using threshold cryptography techniques. The advantage of this approach is each revocation server may have its own standard that must be met, reducing the power of any one authority.
- **Reputation-based.** In this scheme a nodes filtering key is added to the IWL when it has a negative reputation. Initially, all nodes start with a positive reputation, which is decremented for each malicious action. The benefit of this approach is several nodes are involved in deciding what is malicious and what isn't. For example, the Craigslist website is able to process about 10 million new Internet ads each month using only a handful of staff because the system uses a reputation-based system for self-regulation. The system is self-regulation because any user can tag a post as "inappropriate", and if enough users tag a particular post it is pulled for further review. Will the success of Craigslist carry over to this system? One problem is we currently do not know how to create a reputation system that prevents "framing" attacks where a coalition of users can falsely accuse a legitimate node of being malicious. However, there is ongoing research in making reputation systems secure.

**Challenge: Removing Nodes from the IWL.** How do we remove a node from the IWL? The mechanism is straightforward: simply re-run the `Initialize` procedure, giving the node a new identity. Of course IA's should verify that the computer is free of infections, e.g., via a security scan.

**Challenge: Unresponsive or Malicious IA's.** A difficult problem is how to deal with misbehaving IA's. Since each packet is signed, we know which IA is misbehaving. The IA may be misbehaving in several ways: it may be minting new identities for compromised computers, failing to publish filtering keys for malicious nodes, etc.

Note that an end-node can always tell which IA is responsible for a signature, thus use local policy to decide when an IA is acting inappropriately. However, this isn't a full solution: a large IA may be unresponsive but filtering is impossible. In the current Internet architecture, this problem also appears: e.g., "bulletproof" ISP's that are spammer-friendly. In the current architecture, we rely on courts rather than technology to decide how to handle such authorities. However, we can do better. For example, threshold cryptography can be used such that a coalition of federated IA's can decide when another IA is misbehaving and revoke all privileges.

At the end of the day, nodes are likely to trust someone. After all, individuals trust banks, hospitals, Verisign, etc. The question is how much trust should be put in each authority, and what repercussions are there for misbehaving authorities.

## 5 Internet Reputation Systems

While Internet Watch Lists is a binary decision system (i.e., an identity is either on the list or not), we could generalize the system into an Internet Reputation System, where an identity is assigned a score showing its reputation. Thus, a party involved in a communication could make better informed decision about its action based on the other party's reputation. A reputation function could be based on the identity's behavior and history or even one's lack of history. For example, a short-lived website tends to have a higher probability of being a phishing or malicious website; a newly announced route may have a higher probability of being a misconfiguration or a malicious routing attack; a new connection contact may have a higher probability of being a scanner or a probe. More than one factor can be taken into consideration. For example, a new node on a well maintained network is less likely to be malicious than a new node on a spammer-friendly network [1]. The more factors incorporated, the more accurate and representative the reputation system can be, as it takes more facets of an identity into account.

Such a reputation system could allow one to make better informed decisions. For example, certain security defenses are expensive and thus may not be suitable to apply all the time. A reputation system could serve as a guidance for resource allocation, enabling the more expensive security defenses to be applied on actions involving identities with lower reputation.

However, many research challenges remain to make a really effective and viable Internet Reputation System. Many of these challenges are shared with the Internet Watch List approach: a viable Internet Reputation System should be practical, secure, and allow for policy diversity. In particular, the system should support incremental deployment; should be secure, especially against malicious attacks aiming to manipulate the scores in the reputation system to either hide malicious activities or frame legitimate identities; should scale to Internet size and be efficient; should allow federation and decentralized cooperation; and should protect users' privacy. Due to space limit, we do not further elaborate here.

## 6 Conclusion

Attackers who control thousands to millions of hosts pose a significant security threat regardless of the specifics of the attack employed. Therefore, we need to build attack-agnostic defense mechanisms. We have shown an Internet Watch List as one extreme design point for attack-agnostic defense. We have also provided a construction for the watch list that allows anyone to filter a compromised host while retaining anonymity and privacy of non-compromised hosts. The IWL highlights many of the trade-offs for attack-agnostic approaches. We hope this work will spark interest in the research community for more attack-agnostic approaches which can defend against many different kinds of attacks.

## Acknowledgment

The authors would like to thank the anonymous reviewers and Lujo Bauer for their valuable comments and suggestions.

## References

[1] `http://www.senderbase.org`.

[2] D-shield. `dshield.org`.

[3] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *Crypto 2000*, 2000.

[4] H. Ballani, Y. Chawathe, S. Ratnasamy, T. Roscoe, and S. Shenker. Off by default! In *HotNets*, 2005.

[5] S. Bellovin, D. Clark, A. Perrig, and D. Song. A clean slate design for the next generation secure internet. `http://sparrow.ece.cmu.edu/~adrian/projects/ngsi.pdf`, 2006.

[6] R. Bhattacherjee and A. Goel. Avoiding ballot stuffing ebay-like reputation systems, 2005.

[7] D. Boneh and H. Shacham. Group signatures with verifier-local revocation. In *ACM Conference on Computer and Communications Security (CCS)*, 2004.

[8] A. Cheng and E. Friedman. Sybilproof reputation mechanisms. In *Proc. of ACM SIGCOMM Workshop on Economics of peer-to-peer systems*, 2005.

[9] D. Dagon, C. Zou, and W. Lee. Modeling botnet propagation using time zones. In *Proc. of the 13th Annual Network and Distributed System Security Symposium (NDSS'06)*, 2006.

[10] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson. Host identity protocol. `http://homebase.htt-consult.com/Docs/draft-ietf-hip-base-02.txt`, 2005.

[11] B. Patil and G. D. (chairs). Mobility for ipv6 (mip6). `http://www.ietf.org/html.charters/mip6-charter.html`.

[12] D. Song. Practical forward secure group signature schemes. In *8th ACM Conference on Computer and Communications Security (CCS-8)*, 2001.

[13] K. Walsh and E. G. Sirer. Fighting peer-to-peer spam and decoys with object reputation, 2005.

[14] A. Yaar, A. Perrig, and D. Song. SIFF: An endhost capability mechanism to mitigate DDoS flooding attacks. In *Proc. of IEEE Symposium on Security and Privacy (Oakland)*, 2004.

[15] H. Yu, J. Rexford, and E. Felten. A distributed reputation approach to cooperative internet routing protection. In *Proc. of Workshop on Secure Network Protocols (NPSEC)*, 2005.