TU/e technische universiteit eindhoven

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Chosen-prefix collisions for MD5 and colliding X.509 certificates for different identities

Marc Stevens (TU/e)

Arjen Lenstra (EPFL)

Benne de Weger (TU/e)

What could be done before?

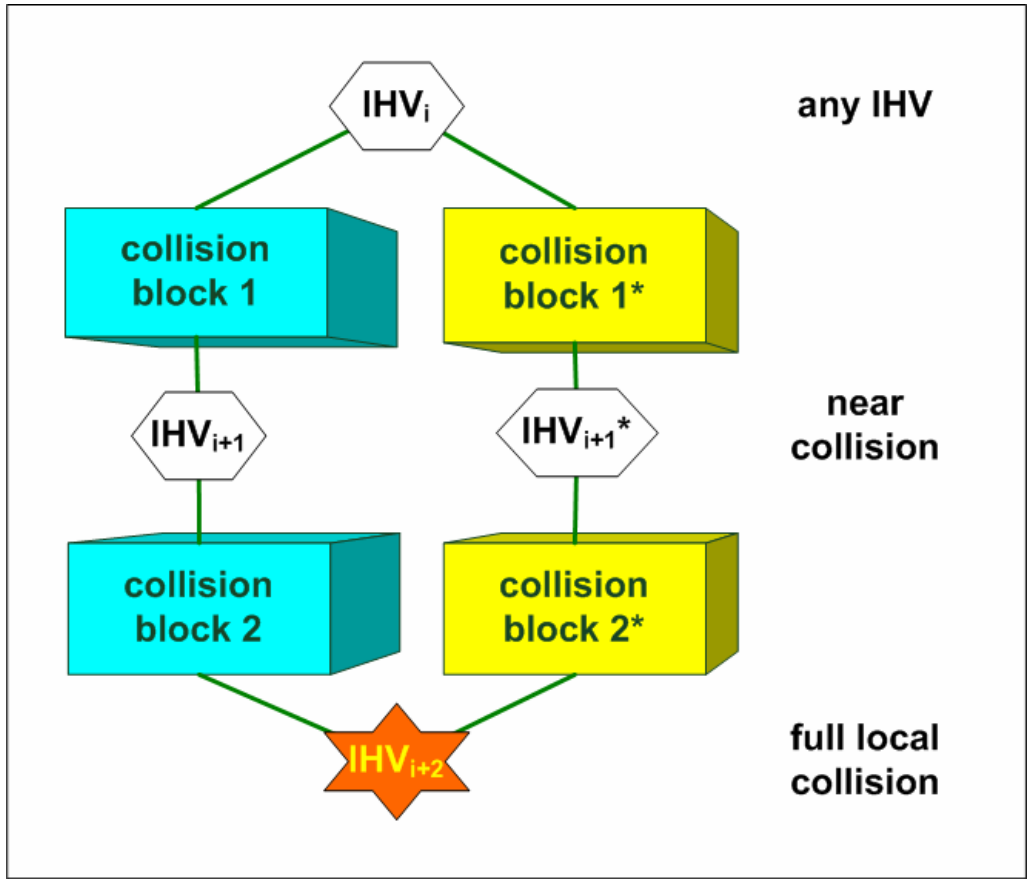
Wang et al. 2004:

MD5 collision

for *identical* IHVs

Complexity:

Klima and MS 2006:
2^{27} MD5 compressions
(seconds on a PC)



What is new in our paper?

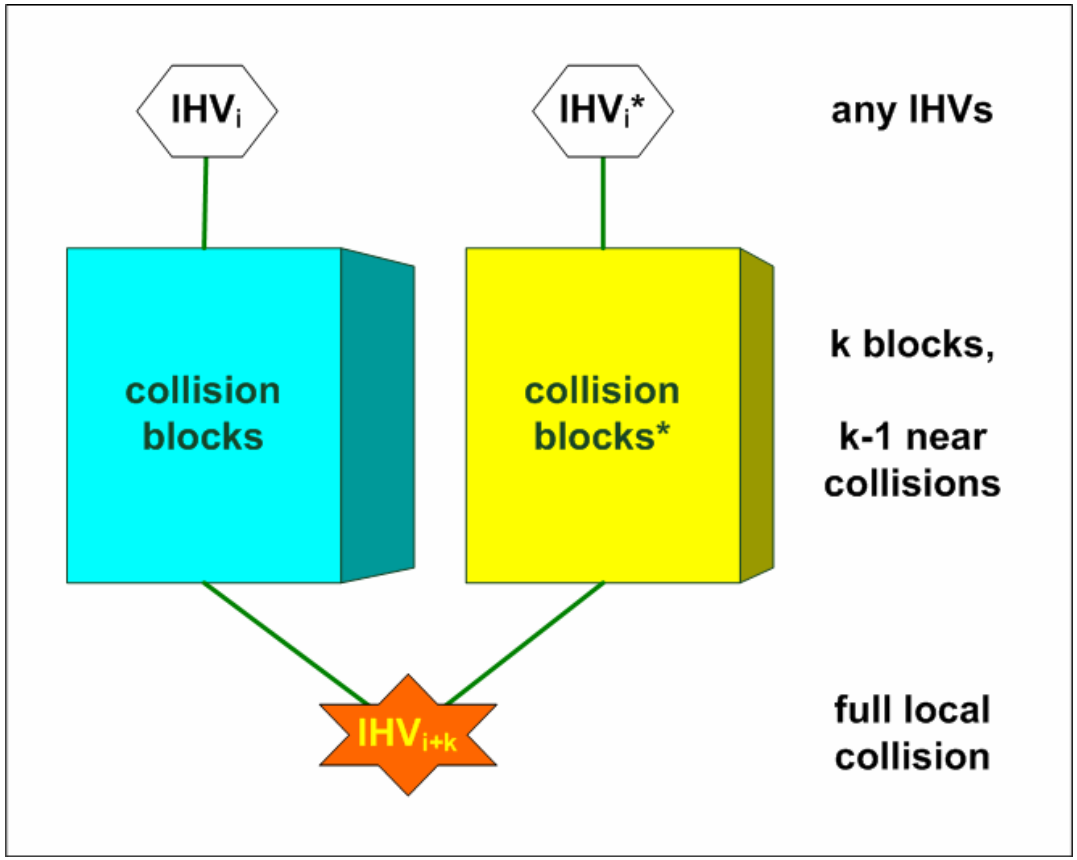
MD5 collision

for *different* IHVs

Complexity:

- Estimated 2^{52} MD5 compressions
- (Months using parallelization)

One example available



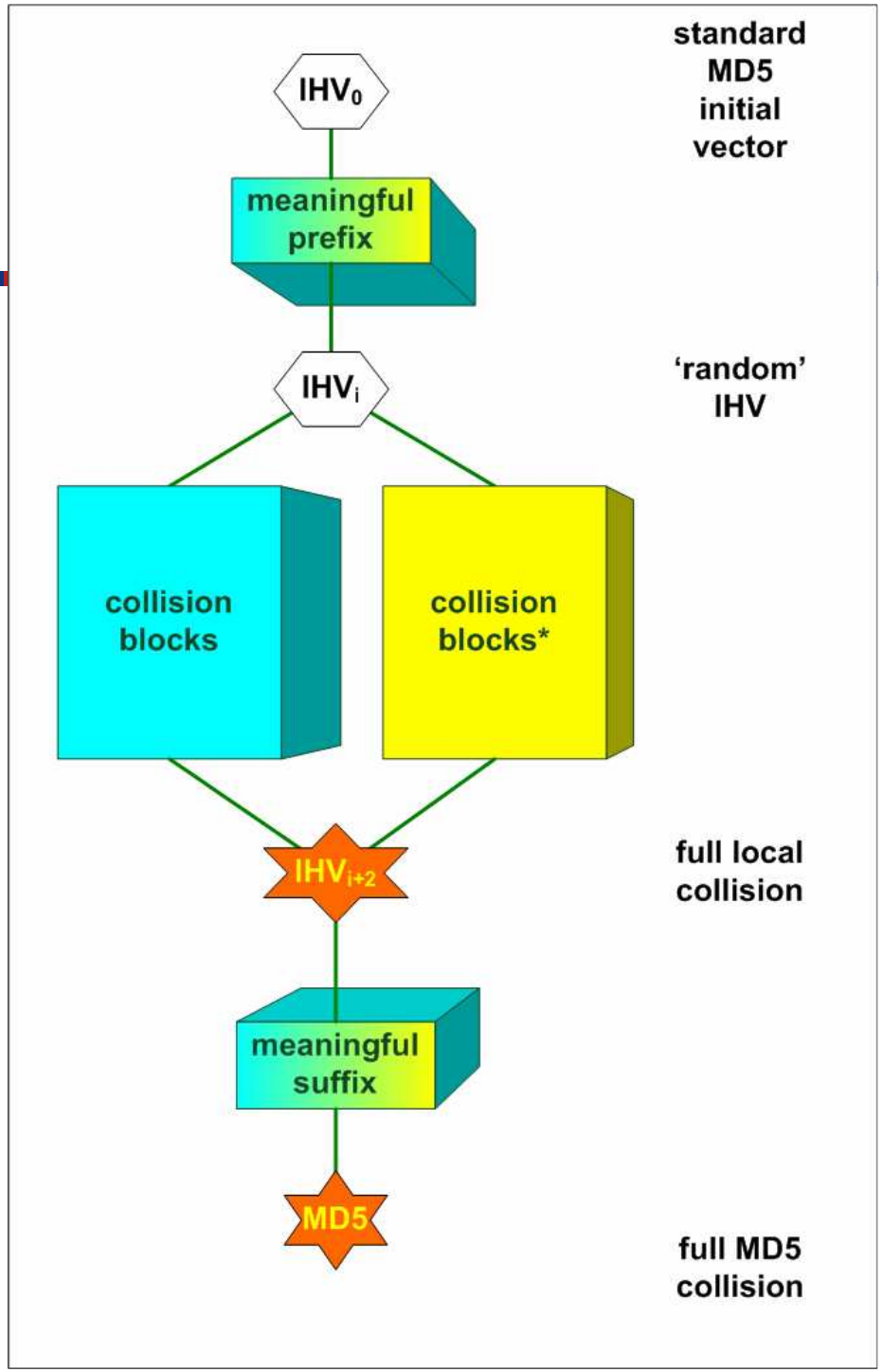
Meaningful Wang-collision

Wang's construction requires

- Identical prefixes
- Identical suffixes

Keep meaningfulness:

- Collision blocks are random-looking
- Must be hidden in meaningful data



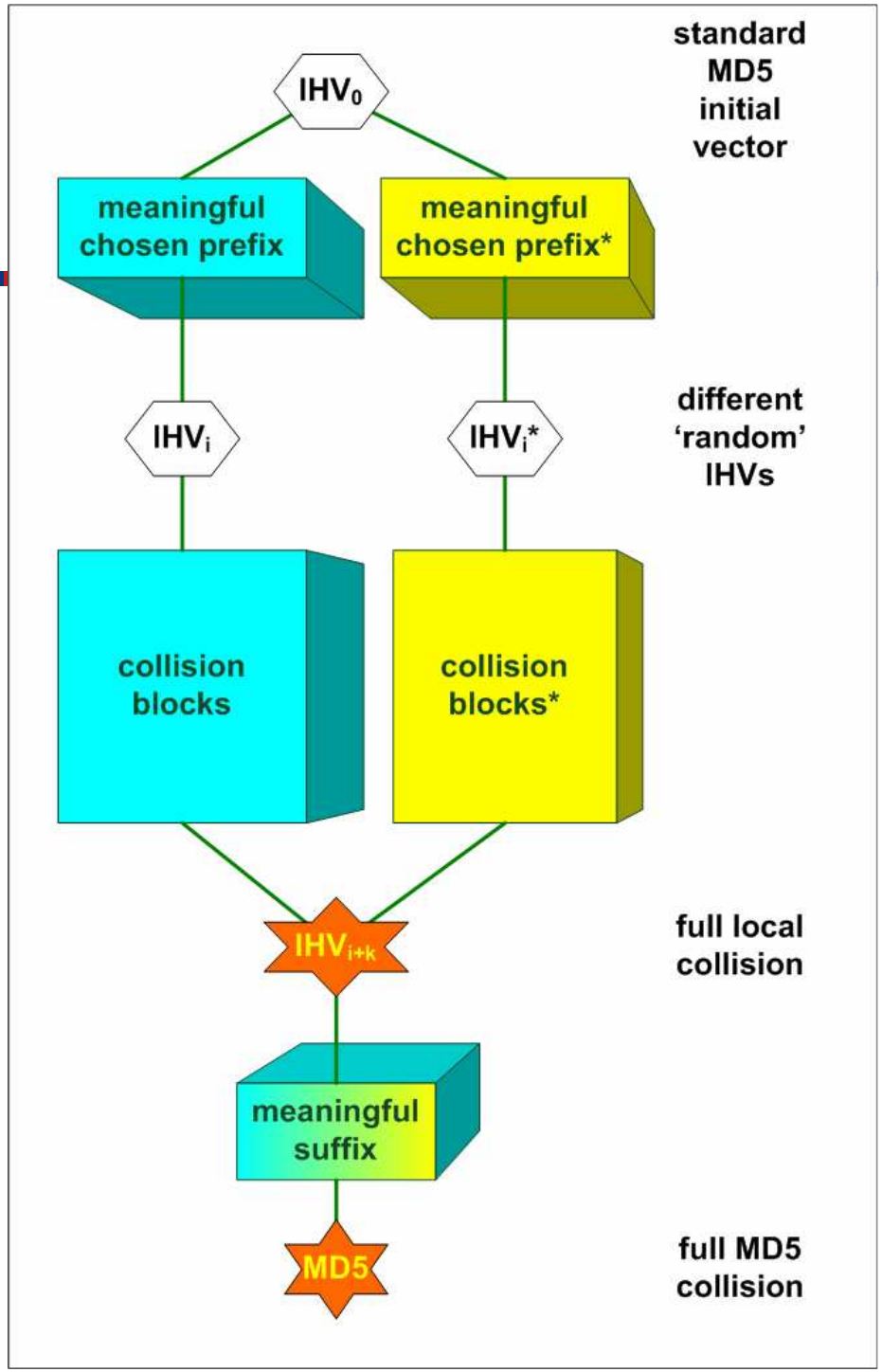
Chosen-prefix collision

Our construction allows

- different prefixes chosen at will (same length)

Still requires identical suffixes

Still need to hide collision in meaningful data



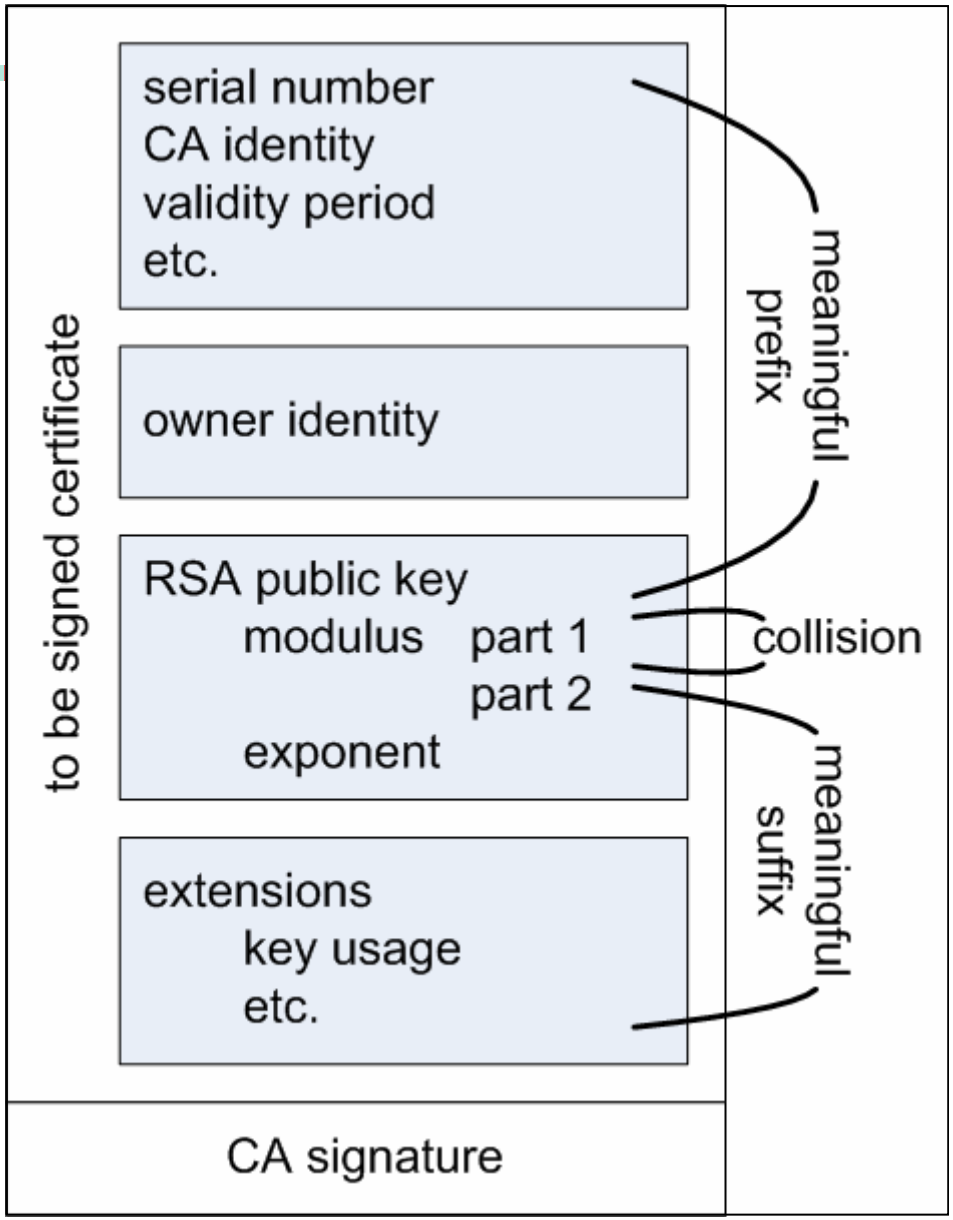
Colliding X.509 certificates

X.509 Certificate contains
public key

Perfect hiding place
for collision blocks

AKL-BdW 2005:
Pair of secure RSA-moduli
of the form

$$p_1\, q_1 = b_1 \,||\, b_0$$
$$p_2\, q_2 = b_2 \,||\, b_0$$



Why is this fun (dangerous)?

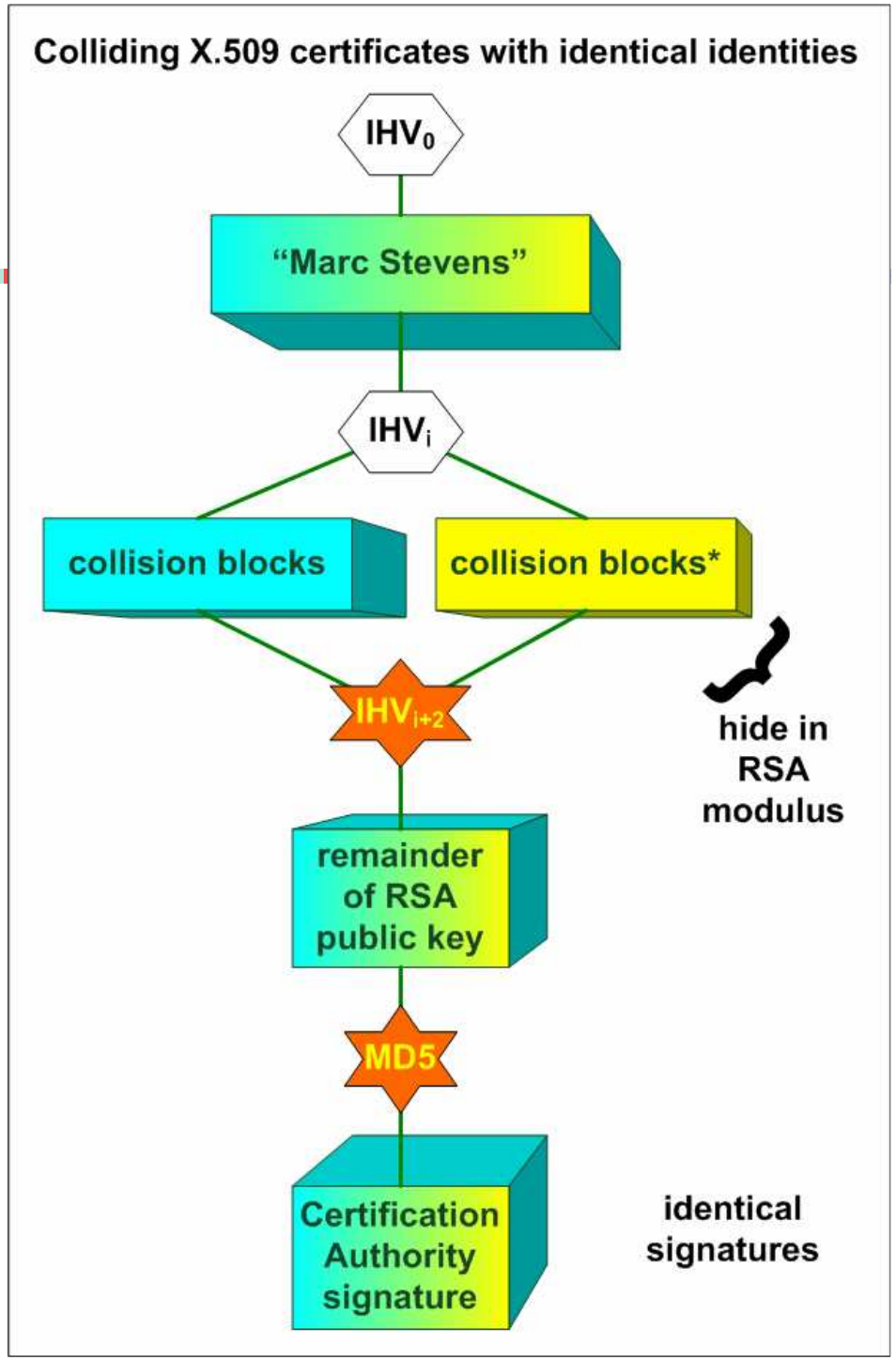
AKL-XW-BdW 2005:

Colliding certificates for *identical* identities

Only different in public keys

No realistic abuse scenario

Nevertheless violation of PKI principles



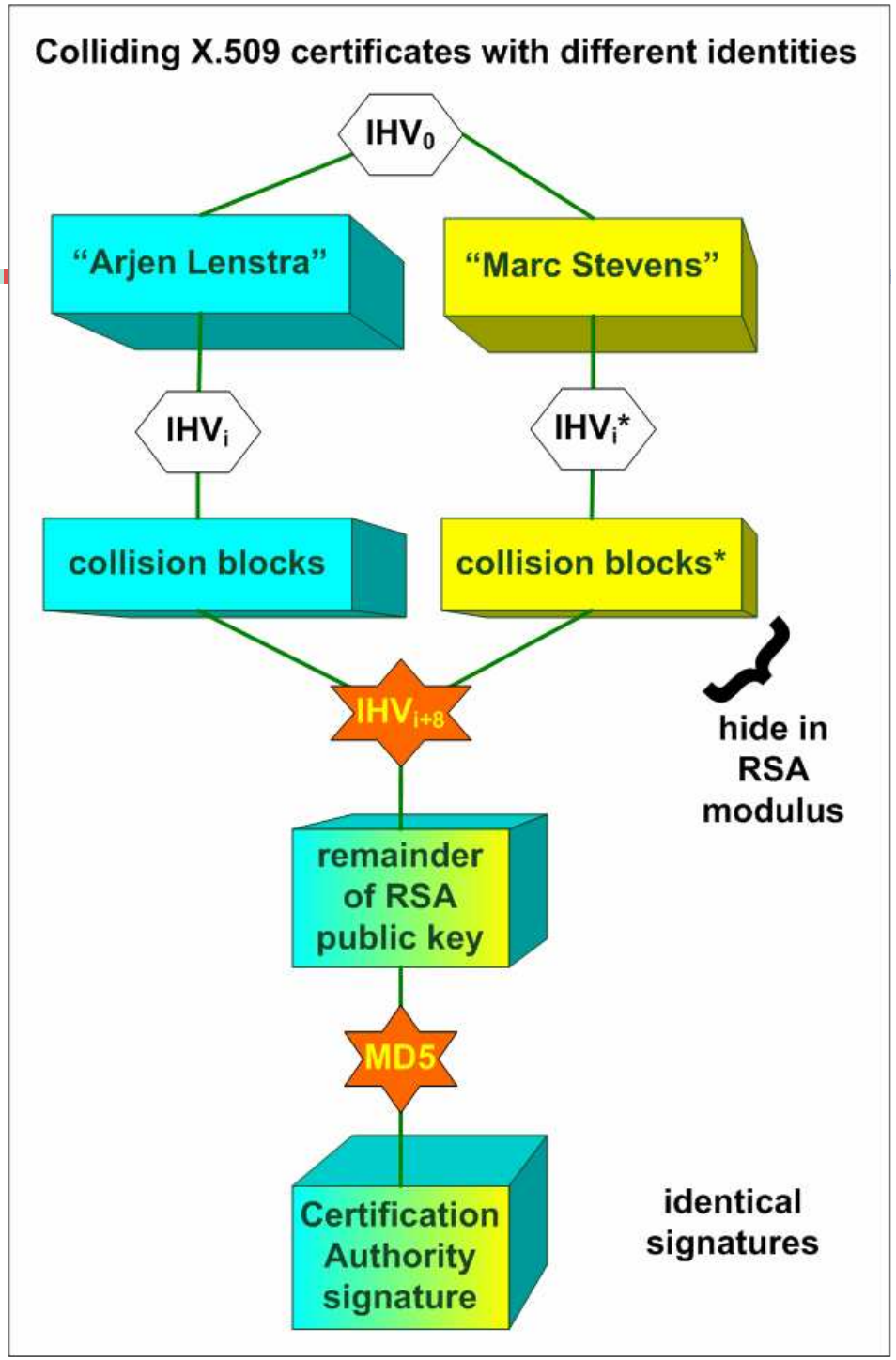
Why is this fun (dangerous)?

This work:

Colliding certificates for
different identities

Our chosen-prefix example
was constructed for
a pair of certificates

Abuse scenarios slightly
more realistic



Why are chosen-prefix collisions more fun?

Daum / Lucks 2005: colliding postscript documents

- Based on Wang's collisions
- Each of the two files contain both documents in full
- Collision difference used in macro to point to which document is to be shown on the screen

We can do better now

using Chosen-prefix collisions

- Each file contains only one document (different)
- Not relying on macro features of postscript
- Need to hide short random looking block
- E.g. in bitmap image

Why are they even more fun?

Mislead download integrity protection

- Given files "**`Word.exe`**" and "**`Worse.exe`**"
- Compute short appendages s.t. resulting files have identical MD5 hash
- Enlarged files are still executable (!)
- Convince code signer to sign enlarged "**`Word.exe`**"
- "**`Worse.exe`**" has same signature
- Publish signed enlarged "**`Worse.exe`**" as "**`Word.exe`**"

Why are they really dangerous?

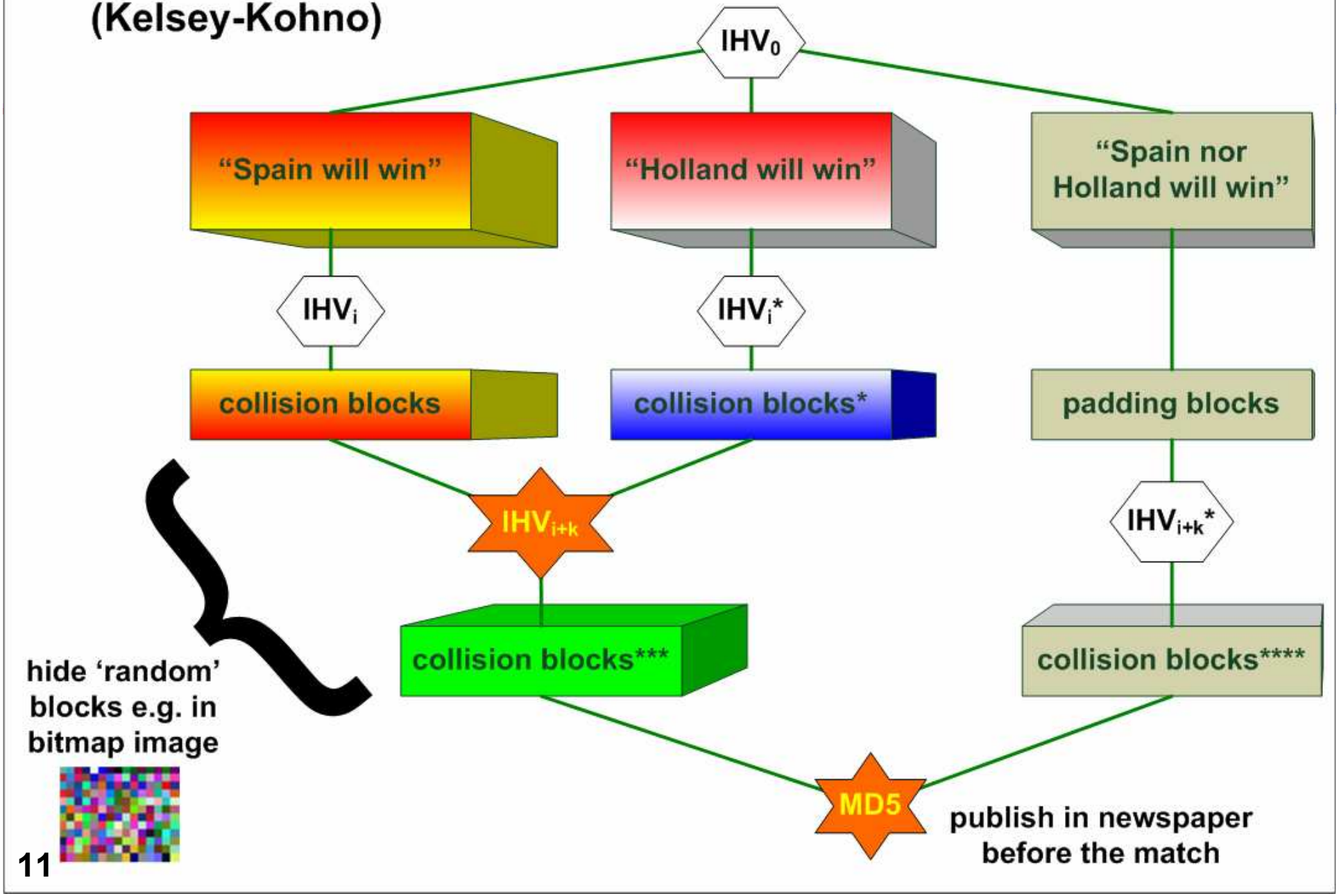
Nostradamus attack (Kelsey-Kohno):

- Predict the next European Soccer Champion
- Construct colliding documents stating “X will win” where X is “Spain” or “Holland” or “Spain nor Holland”
- Commitment: hash value of colliding documents
- After final match show corresponding document (“Holland will win”)
- Cash your bet

This attack is feasible now ☺

only 2 chosen-prefix collisions required

Nostradamus attack (Kelsey-Kohno)



Why are chosen-prefix collisions in many applications not that dangerous?

(and not as much fun either ☹)

- The attacker must choose both prefixes
- The attacker must hide a random looking bit string in a meaningful message
- When in a forensic investigation both inputs are found, fraud is revealed

Mainly:

- This is not a (2nd) pre-image attack

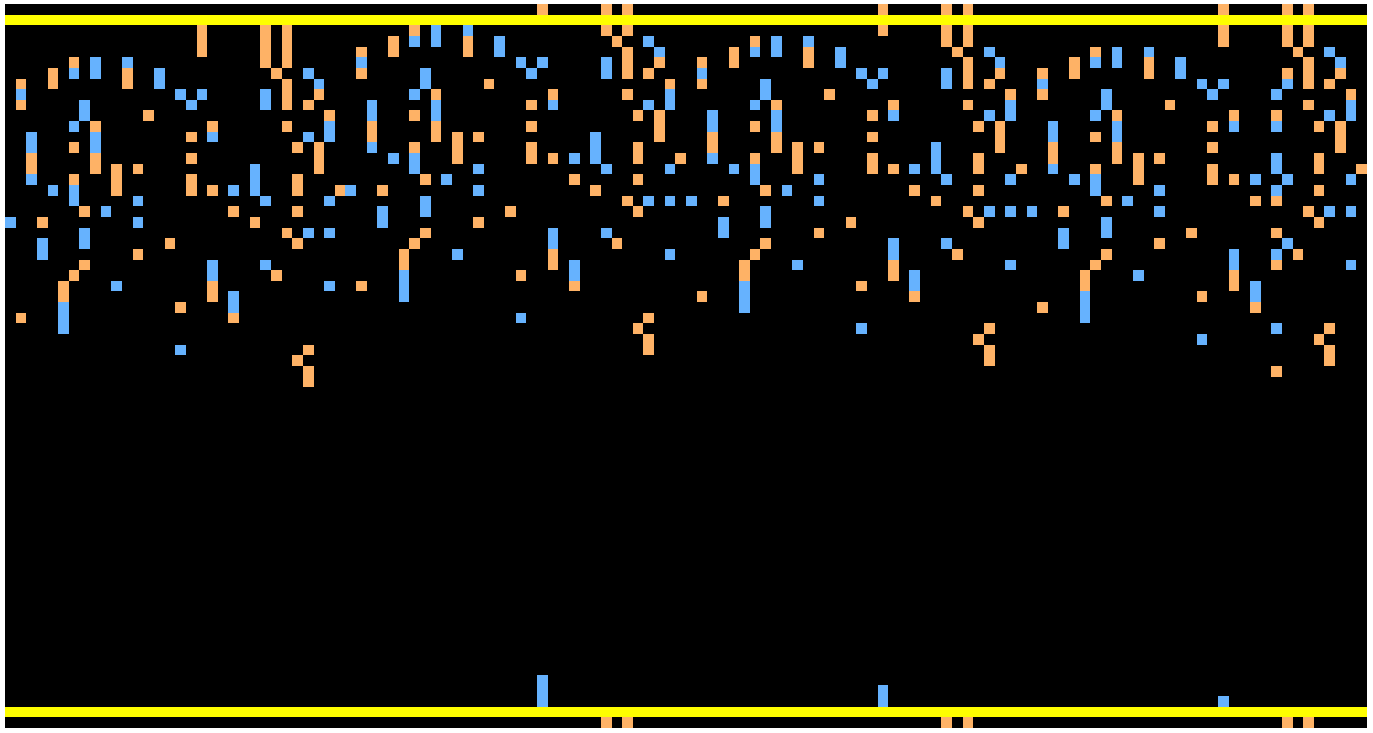
What's new in our techniques?

Algorithm for constructing differential paths (for the compression function)

- Use bitconditions to describe differences and specific bitvalues of working state
- Allows control of differences coming out of boolean function of MD5
- Forward: construction given IHV,IHV* easy
- Backward: construction given last working state differences easy
- Meet in the middle: Efficient algorithm tries to connect both

What's new in our techniques?

- First and last line show IHV_i and IHV_{i+1} differences
- Remaining lines show working state differences



What's new in our techniques?

Algorithm for constructing differential paths (for the compression function)

- For any input IHV,IHV* this has to be done separately
- We fix a family of upper differential paths
- With 1 message block pair we can cancel any triple of bit differences $(0,+2^b,+2^b,+2^b)$
- Not every IHV,IHV* can be eliminated using only these near-collisions

How does this produce collisions?

Do a 96-bit birthday procedure

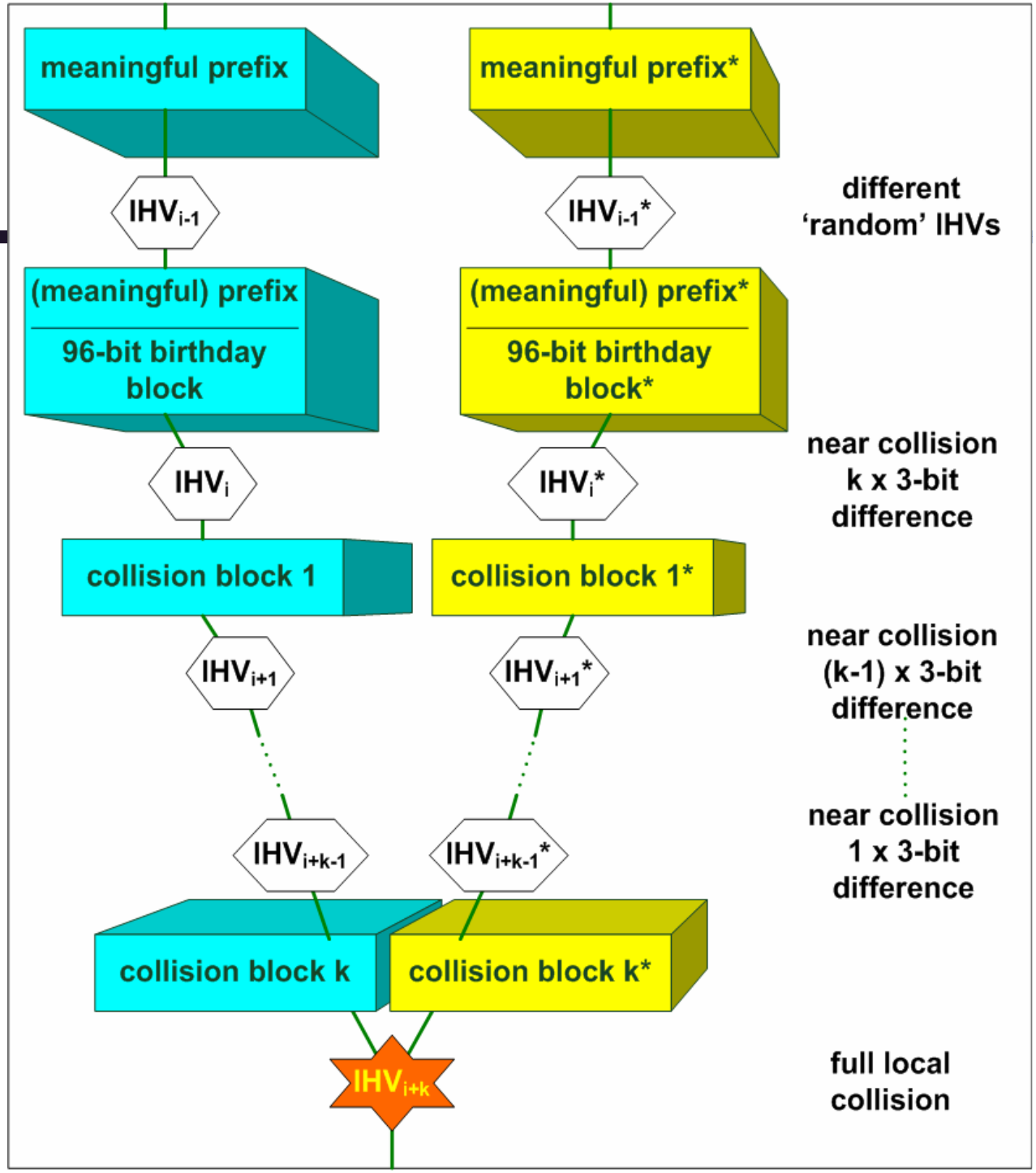
- To get an IHV difference of the form: (0,d,d,d)
- Complexity: 2^{49} MD5 compressions
- Birthday a bit longer to get fewer bit differences
- This is the computational bottleneck

Then cancel triples of bit differences one by one using automated differential path construction

Collision construction

Example shown for k triples of bit differences

Collision length: 96+k*512 bits



Our example of a chosen-prefix collision

Created for colliding X.509 certificates

- 4192 bit collision hidden in 8192 bit RSA moduli
- Certificates verify with standard software (openssl, MS Internet Explorer)

HashClash project

- Used BOINC distributed software
- About 1200 volunteers and cluster at TU/e
- Peak performance: 400 GFlops
- Took about 6 months in total
- Website with certificates and details: www.win.tue.nl/hashclash

The example, please check ☺

MD5 (

Arjen K. Lenstra

prefix

```
30820511A003020102020204010C0001300D06092A864886F70D0101040500303D311A3018060355040313114861736820436F6C6C6973696F6E20434131123010
0603550407130945696E64686F76656E310B3009060355040613024E4C301E170D3036303130313030303030315A170D3037313233313233353935395A305431
193017060355040313104 1726A656E204B2E204C656E73747261 3116301406035504 0A130D436F6C6C6973696F6E6169727331123010060355040713094569 6E
64686F76656E310B3009060355040613024E4C30820422300D06092A864886F70D0101010500038204 0F003082040A0282040100 EE73E7D6B3B34FBAA1393D02
A47425818DC84F86736E907228BBE8770203858D8CF1837AFF5E6C2213036AF3D95C77E9C2237D608CC4A9FB97308BBF9828612F1599E2615BCCDEDA5930532F
B3DD117278E494401433630E7461C1DC9B801B2E552015A513FF7AE7973EF44B8352E4E04979B31EB600654D51F4A381CEBE3F0BD099D130D1456FABE04A3E98
85C8C4FB297B86B57752CD6419809FE37E6286F07732D1E069A5B4E56670B8BBBAE5C211742A131D05711CF1FE32AF933F1EEF224762E3AADAC17C40E448CA41
A879A03D3CF665F239C7F3FE82B384E835E7C9E8BDEE30C268A2121284789DF42F44906F19B79026464436E1DA65FA0C53A377FA0D2B012B7DDC2855DAE5B551
51E28034112120B5E79EC5F26A9F69DA85D74EF6A97A0B1164EFA25FB1AE26BA451CCDA7A2E784339C447D560549A60BF0676294BF580C919EC457025D3C7860
B98296C0AB9FE5B1D353882E26C1F721B41899D972B5A1D5050B684536448010AF8C7AFF7CE8EACCB9B1FBBDD129D4F5D499FB812924DF302CB3C45023386297
9396B3A46CD0FF7F1426711C459297B65D1CEF66C18751E094BF08F3B2981C5CCE52D963D5A4259A64557E4D1B9EFE2D9A516D1E6EC8BB37066825AEA6361660
2BD7D11625A06A90739B4D0A06EA872A3AF9EBA12629BED67940561BD9374A89D60F0D722C9FEB6833EC53F0B0FD76AA047B66C90FCEB1D2E22CC099B9A4B93E
0000000F54A895176E4C295A405FAF54CEE82D043A45CE40B155BE34EBDE784785A25B7F894D424FA127B157A8A120F99FE53102C81FA90E0B9BDA1BA775DF75
D9152A80257A1ED352DD49E57E068FF3F02CABD4AC97DBBC3FA0205A74302F65C7F49A419E08FD54BFAFC14D78ABAAB30DDB3FC848E3DF02C5A40EDA248C9FF4
7482850CFDFBDD9BC55547B7404F5803C1BB81632173127E1A93B24AFB6E7A80450865DB374676D576BA5296CCC6C13082D1AB36521F1A8AD945466B9EF06AF4
3A02D70B7FB8B7DC6D268C3DBA6898F6552FA3FBB33DCBFADA7B33FA75D93AFE262BD37AFF75995FD0E9774BA5A26A7C443FF34E461502A2CB777E982D007375
14B88ED28D61F428E88387DF2BF02230AD17A9D44FF364850A07DB42A7826AC2EE3899CAC3EC274721D476D96658F53716676587F8FF14DB8DE6741AFA2206DB
A3B11828BA87C6E1E88A022F1AA8DDD037EAB049B5C7D3053D0A63D7861DEA07B3D8B720DE068CF47E657BB44450B85D52F749D59572DF0C0E3433B47C9AA19A
856F1DC3CDADBAFB143035C85A53AF5722038F765C0D621B66B69FFFFD091D4A661A453BF1DAED1A3A2341B37D7F623B158F6EC02B49A25364430FCB5861483E
1E9543ED2EE7E54A4C108A6E641940980EE60D14AEE559AF30037E75B2309CE021FFE3109BF2053892AB0AE403516E2AB58067F70203010001A31A3018300906
03551D1304023000300B0603551D0F0404030205E0
```

coll

suffix

) = MD5 (

Marc Stevens

prefix

```
30820511A003020102020204020C0001300D06092A864886F70D0101040500303D311A3018060355040313114861736820436F6C6C6973696F6E20434131123010
0603550407130945696E64686F76656E310B3009060355040613024E4C301E170D3036303130313030303030315A170D3037313233313233353935395A305431
153013060355040313 0C4D617263205374657 6565 6E73 311A3018060355040A1311436F6C6C6973696F6E20466163746F72793112301006035504071309456 96E
64686F76656E310B3009060355040613024E4C30820422300D06092A864886F70D0101010500038204 0F003082040A0282040100 1A09B4CB40C7267AAF017F9B
A47425818DC84F86736E907228BBE8770203858D8CF1837AFF5E6C2213036AF3D95C77E9C2237D608CC4A9FB97307BBF9828612F1599E2615BCCDEDA5930532F
B3DD117278E494401433630E7461C1DC9B801B2E552015A513FF7AE7973EF44B8352E4E04979B31EB600654D51F4A481CEBE3F0BD099D130D1456FABE04A3E98
85C8C4FB297B86B57752CD6419809FE37E6286F07732D1E069A5B4E56670B8BBBAE5C211742A131D05711CF1FE22AF933F1EEF224762E3AADAC17C40E448CA41
A879A03D3CF665F239C7F3FE82B384E835E7C9E8BDEE30C268A2121284789DF42F44906F19B79026464436E1DA64FA0C53A377FA0D2B012B7DDC2855DAE5B551
51E28034112120B5E79EC5F26A9F69DA85D74EF6A97A0B1164EFA25FB1AE26BA451CCDA7A2E784339C447D562549A60BF0676294BF580C919EC457025D3C7860
B98296C0AB9FE5B1D353882E26C1F721B41899D972B5A1D5050B684536448010AF8C7AFF7CE8EACCB9B1FBBDC929D4F5D499FB812924DF302CB3C45023386297
9396B3A46CD0FF7F1426711C459297B65D1CEF66C18751E094BF08F3B2981C5CCE52D963D5A4259A64557E4D1B9EFE0D9A516D1E6EC8BB37066825AEA6361660
2BD7D11625A06A90739B4D0A06EA872A3AF9EBA12629BED67940561BD9374A89D60F0D722C9FEB6833EC53F0B0FD76A2047B66C90FCEB1D2E22CC099B9A4B93E
0000000F54A895176E4C295A405FAF54CEE82D043A45CE40B155BE34EBDE784785A25B7F894D424FA127B157A8A120F99FE53102C81FA90E0B9BDA1BA775DF75
D9152A80257A1ED352DD49E57E068FF3F02CABD4AC97DBBC3FA0205A74302F65C7F49A419E08FD54BFAFC14D78ABAAB30DDB3FC848E3DF02C5A40EDA248C9FF4
7482850CFDFBDD9BC55547B7404F5803C1BB81632173127E1A93B24AFB6E7A80450865DB374676D576BA5296CCC6C13082D1AB36521F1A8AD945466B9EF06AF4
3A02D70B7FB8B7DC6D268C3DBA6898F6552FA3FBB33DCBFADA7B33FA75D93AFE262BD37AFF75995FD0E9774BA5A26A7C443FF34E461502A2CB777E982D007375
14B88ED28D61F428E88387DF2BF02230AD17A9D44FF364850A07DB42A7826AC2EE3899CAC3EC274721D476D96658F53716676587F8FF14DB8DE6741AFA2206DB
A3B11828BA87C6E1E88A022F1AA8DDD037EAB049B5C7D3053D0A63D7861DEA07B3D8B720DE068CF47E657BB44450B85D52F749D59572DF0C0E3433B47C9AA19A
856F1DC3CDADBAFB143035C85A53AF5722038F765C0D621B66B69FFFFD091D4A661A453BF1DAED1A3A2341B37D7F623B158F6EC02B49A25364430FCB5861483E
1E9543ED2EE7E54A4C108A6E641940980EE60D14AEE559AF30037E75B2309CE021FFE3109BF2053892AB0AE403516E2AB58067F70203010001A31A3018300906
03551D1304023000300B0603551D0F0404030205E0
```

coll

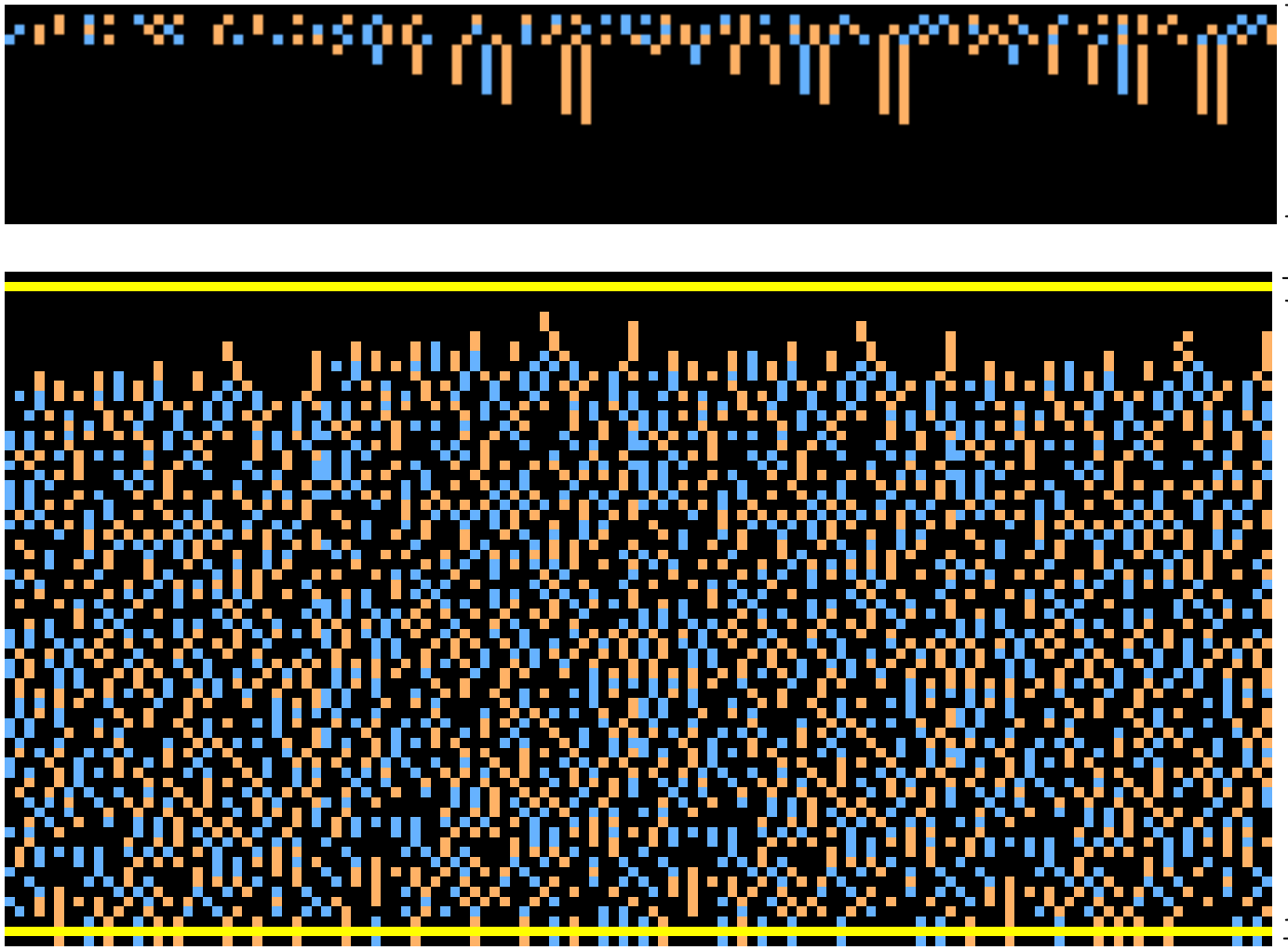
suffix

) = C6B2FE88912770FC6F2DB71F58C7D251

The example (zoomed in)

```
MD5Compress(    2D857B4E0479B7259F7662D47771220B,
    A47425818DC84F86736E907228BBE8770203858D8CF1837AFF5E6C2213036AF3D95C77E9C2237D608CC4A9FB
    973088BF9828612F1599E2615BCCDEDA5930532FB3DD117278E494401433630E7461C1DC9B801B2E552015A5
    13FF7AE7973EF44B8352E4E04979B31EB600654D51F4A381CEBE3F0BD099D130D1456FABE04A3E9885C8C4FB
    297B86B57752CD6419809FE37E6286F07732D1E069A5B4E56670B8BBBAE5C211742A131D05711CF1FE32AF93
    3F1EEF224762E3AADAC17C40E448CA41A879A03D3CF665F239C7F3FE82B384E835E7C9E8BDEE30C268A21212
    84789DF42F44906F19B79026464436E1DA65FA0C53A377FA0D2B012B7DDC2855DAE5B55151E28034112120B5
    E79EC5F26A9F69DA85D74EF6A97A0B1164EFA25FB1AE26BA451CCDA7A2E784339C447D5605549A60BF0676294
    BF580C919EC457025D3C7860B98296C0AB9FE5B1D353882E26C1F721B41899D972B5A1D5050B684536448010
    AF8C7AFF7CE8EACCB9B1FBBDD129D4F5D499FB812924DF302CB3C450233862979396B3A46CD0FF7F1426711C
    459297B65D1CEF66C18751E094BF08F3B2981C5CCE52D963D5A4259A64557E4D1B9EFE2D9A516D1E6EC8BB37
    066825AEA63616602BD7D11625A06A90739B4D0A06EA872A3AF9EBA12629BED67940561BD9374A89D60F0D72
    2C9FEB6833EC53F0B0FD76AA047B66C90FCEB1D2E22CC099B9A4B93E
) =
MD5Compress(    2D857B4EA419FB613F17A61017126647,
    A47425818DC84F86736E907228BBE8770203858D8CF1837AFF5E6C2213036AF3D95C77E9C2237D608CC4A9FB
    973077BBF9828612F1599E2615BCCDEDA5930532FB3DD117278E494401433630E7461C1DC9B801B2E552015A5
    13FF7AE7973EF44B8352E4E04979B31EB600654D51F4A481CEBE3F0BD099D130D1456FABE04A3E9885C8C4FB
    297B86B57752CD6419809FE37E6286F07732D1E069A5B4E56670B8BBBAE5C211742A131D05711CF1FE22AF93
    3F1EEF224762E3AADAC17C40E448CA41A879A03D3CF665F239C7F3FE82B384E835E7C9E8BDEE30C268A21212
    84789DF42F44906F19B79026464436E1DA64FA0C53A377FA0D2B012B7DDC2855DAE5B55151E28034112120B5
    E79EC5F26A9F69DA85D74EF6A97A0B1164EFA25FB1AE26BA451CCDA7A2E784339C447D562549A60BF0676294
    BF580C919EC457025D3C7860B98296C0AB9FE5B1D353882E26C1F721B41899D972B5A1D5050B684536448010
    AF8C7AFF7CE8EACCB9B1FBBDC929D4F5D499FB812924DF302CB3C450233862979396B3A46CD0FF7F1426711C
    459297B65D1CEF66C18751E094BF08F3B2981C5CCE52D963D5A4259A64557E4D1B9EFE0D9A516D1E6EC8BB37
    066825AEA63616602BD7D11625A06A90739B4D0A06EA872A3AF9EBA12629BED67940561BD9374A89D60F0D72
    2C9FEB6833EC53F0B0FD76A2047B66C90FCEB1D2E22CC099B9A4B93E
) = 505D9746FAB00B328018DBC34A87DF11
```

Visualisations of the example



Conclusion

- Creating chosen-prefix collisions for MD5 is very feasible
- Going deeper into the grey area between collision resistance and (2^{nd}) preimage resistance
- Attack scenarios questionable
  - though Nostradamus attack sounds good
- Might help convince users that MD5 is dead and should be buried
- Ongoing work, to be expected soon:
  - performance improvements
  - fully automated software